

Jet Propulsion Laboratory
California Institute of Technology

Design and Development of Physical Spacecraft Systems using GenAI

a NASA PESTO Study

Ryan T. Watkins
ryan.t.watkins@jpl.nasa.gov

Kalind Carpenter
kalind.carpenter@jpl.nasa.gov

Christine A. Gebara
Christine.A.Gebara@jpl.nasa.gov

Jacob Izraelevitz
jacob.izraelevitz@jpl.nasa.gov

Kelli J. McCoy
Kelli.J.Mccoy@jpl.nasa.gov

Rezwanur Rahman
rezwanur.rahman@jpl.nasa.gov

Jefferey Hall
jeffery.l.hall@jpl.nasa.gov

Brian Lim
brian.y.lim@jpl.nasa.gov

August 24, 2026

© 2026 California Institute of Technology. Government sponsorship acknowledged.

Table of contents

Executive Summary 6
1 Introduction 12
2 GenAI Technology Landscape 14
2.1 What Makes This AI Different? 14
2.1.a Traditional AI: Specialized Tools for Specific Tasks 14
2.1.b The Generative AI Paradigm Shift 14
2.2 Large Language Models: Capabilities and Limitations 15
2.2.a How LLMs Work: A High-Level View 15
2.2.b Core Capabilities 17
2.2.c Fundamental Limitations 17
2.3 Retrieval Augmented Generation: Connecting LLMs to Knowledge 18
2.3.a How RAG Works 19
2.3.b Key Benefits for NASA and General Engineering Applications 20
2.3.c Limitations and Challenges 20
2.4 Beyond Chat: Agentic Frameworks 21
2.4.a Tool Use and Function Calling 21
2.4.b Autonomous Agents 23
2.4.c Multi-Agent Systems 23
3 Assessment Study 25
3.1 Study Design 25
3.1.a Industry Engagement 25
3.1.b Research and Analysis Methods 25
3.1.c Study Participants and Scope 26
3.2 State of Practice in Aerospace 27
3.2.a Infrastructure 27
3.2.b People 28
3.2.c Institutional Culture 30
3.2.d Strategy 30
3.3 Software Ecosystem Landscape 31
3.3.a Foundation Models Landscape 31
3.3.b Development Infrastructure and Frameworks 32
3.3.c Domain-Specific Tools 32
3.3.d Ecosystem Maturity and Evolution 37
3.3.e Critical Gaps and Limitations 38
3.4 Application Areas 40
3.4.a Use Case Patterns 41
3.4.b Use Case Examples 41
4 Pilot Demonstration Study 60
4.1 Pilot Design 60
4.1.a Case Study Definition: Design Scenario 60
4.1.b Case Study Selection Rationale 60
4.1.c Tool Selection and Configuration 61
4.1.d Approach 61
4.2 Results and Discussion 61

© 2026 California Institute of Technology. Government sponsorship acknowledged.

4.2.a Demonstrated Capabilities 62
4.2.b Limitations and Challenges 66
4.2.c Effective Practices 67
5 Recommendations and Path Forward 70
5.1 Strategic Investment Principles 70
5.1.a Leverage Rather Than Duplicate 70
5.1.b Match Strategy to Application Scale 70
5.2 Specific Recommendations 71
5.2.a NASA-Wide AI Exploration Event 71
5.2.b Public Benchmarks and Challenge Problems 73
5.2.c NASA Knowledge Source Integration 74
5.2.d NASA-Wide Agent Platform 76
5.2.e Dedicated Flight Project GenAI Consultant 77
5.2.f Large-Scale Demonstration Project 78
6 Conclusions 80
7 Acknowledgements 81
Bibliography 81

© 2026 California Institute of Technology. Government sponsorship acknowledged.

List of findings

- **Finding 1**: GenAI systems can operate under ambiguity, accepting intent and constraints rather than requiring exhaustive specifications—a fundamental shift from traditional conditional computing.
- **Finding 2**: RAG systems are essential for GenAI in engineering applications, not only grounding generated content in authoritative knowledge bases but also providing traceability to the source material—a critical component for establishing trust.
- **Finding 3**: Integration of tool calling within AI agents is essential for engineering applications, enabling LLMs to autonomously invoke calculators, simulations, and other engineering tools—grounding reasoning in ground truth physics and math.
- **Finding 4**: NASA's provisioning of GenAI tools spanning multiple data sensitivity levels (including ITAR and EAR data) has been effective and essential to the current degree of workforce adoption and organizational success with these technologies.
- **Finding 5**: While NASA's institutional knowledge sources—lessons learned, standards, technical documentation—remain largely unintegrated with GenAI systems to date, this presents a high-value opportunity, particularly as RAG-based knowledge access is among the most readily implemented capabilities and NASA's knowledge base is a core organizational strength.
- **Finding 6**: The majority of GenAI applications at NASA currently emerge from grassroots innovation rather than top-down institutional strategy.
- **Finding 7**: While NASA has initiated a few top-down strategic programs for GenAI in spacecraft design, greater coordinated institutional effort and investment will be required to achieve advanced capabilities such as automated spacecraft system design.
- **Finding 8**: AI coding assistants represent the most sophisticated and widely adopted GenAI application beyond chatbots, with proven functionality ready for mainstream use across engineering disciplines—not merely for software engineers but as cross-cutting tools valuable for scripting, automation, and document preparation.
- **Finding 9**: AI-based MBSE platforms exhibit the clearest near-term benefit and maturity among engineering-specific GenAI tools, overcoming the traditional overhead and complexity barriers that limited MBSE adoption while proving particularly valuable for early project phases (Phase A and Phase B) and enabling dynamic capabilities beyond classical systems engineering tasks.
- **Finding 10**: Limited CAD native integration and spatial reasoning capabilities represent the most significant shortcoming in current LLM applications for mechanical engineering and other disciplines directly tied to 3D geometries, fundamentally constraining GenAI effectiveness for core engineering tasks.
- **Finding 11**: GenAI coding agents are highly effective at automating preliminary engineering analysis, including calculations for sizing and performance prediction, along with the ability to rapidly create dashboards that provide a more intuitive interface to the interactions of various elements of a system or design.
- **Finding 12**: Systems engineering tasks—requirements derivation, management, and analysis—represent readily achievable GenAI applications, as these language-based activities require contextual understanding across diverse sources, a capability where LLMs excel.
- **Finding 13**: The combination of current tool maturity levels and high accuracy requirements for engineering applications necessitates extensive verification of GenAI outputs—while RAG systems and agent tool usage provide grounding, verification and validation remains a significant and non-trivial workload.

© 2026 California Institute of Technology. Government sponsorship acknowledged.

- **Finding 14**: GenAI-accelerated content generation demands rigorous version control extending beyond software to all engineering data products, as AI produces design changes faster than humans can fully oversee.
- **Finding 15**: Significant GenAI development investment already exists across broader industry in areas where NASA duplication offers minimal return—NASA must strategically identify ecosystem gaps critical for NASA applications that will not emerge naturally from current commercial investments.

List of recommendations

- **Recommendation 1**: Organize a comprehensive, agency-wide AI exploration event delivered as a hybrid program with virtual sessions accessible across all NASA centers supplemented by local in-person activities.
- **Recommendation 2**: Develop and publicly host NASA-specific datasets, benchmarks, and challenge problems that address gaps in public resources while meeting agency evaluation needs.
- **Recommendation 3**: Develop and deploy MCPs providing programmatic access to critical institutional knowledge sources and research literature, enabling both institutional agent platforms and grassroots applications to leverage decades of accumulated technical expertise.
- **Recommendation 4**: Mature and expand existing institutional agent platforms to build substantial user adoption and capability breadth across NASA.
- **Recommendation 5**: Establish a funded position within select flight projects specifically dedicated to identifying, prototyping, and integrating GenAI solutions tailored to mission-specific workflows and challenges.
- **Recommendation 6**: Fund a large-scale demonstration project that applies GenAI-augmented workflows to design and build flight hardware for an actual space mission (such as a smallsat technology demonstration mission), substantially exceeding the scope of individual proofs-of-concept.

© 2026 California Institute of Technology. Government sponsorship acknowledged.

Executive Summary

NASA faces mounting pressure to reduce spacecraft development cost and schedule while maintaining technical excellence amid increasingly complex missions and constrained budgets. Generative Artificial Intelligence (GenAI) offers potential pathways to improve efficiency, yet the technology landscape is dominated by extraordinary claims that make it difficult for decision makers to distinguish genuine near-term capabilities from aspirational promises.

This Planetary Exploration Science Technology Office (PESTO) study provides systematic assessment of GenAI specifically for spacecraft design and development in mission Phases A through D, cutting through hype to deliver actionable findings and recommendations. We employed a multimethod approach combining expert interviews across NASA centers, aerospace companies, and GenAI vendors; participation in the Text-to-Spaceship Symposium at Goddard Space Flight Center; and hands-on pilot demonstrations applying GenAI tools to representative spacecraft design problems. The study examined three interconnected dimensions: aerospace industry practice and adoption patterns, software ecosystem maturity and readiness, and application space opportunities spanning near-term deployment to long-term strategic investment.

The following sections present key findings about the current state of practice in applying GenAI to spacecraft development, along with recommendations for how NASA can most effectively leverage currently available technology and invest strategically in future capabilities. The accompanying report provides comprehensive detail including technology background (**Section 2**), assessment of aerospace industry practice (**Section 3.2**), pilot demonstration results (**Section 4**), and detailed recommendations (**Section 5**). While the full report is lengthy, its scope reflects the complexity and nuance of this technology domain—providing necessary context to cut through the often misleading information prevalent in broader media coverage and enabling readers to develop understanding at whatever level of depth serves their decision-making needs.

Key Findings

There are two critical features required in GenAI systems for engineering applications:

Finding 2

Retrieval Augmented Generation (RAG) systems not only ground generated content in authoritative knowledge bases but also provide traceability to the source material—a critical component for establishing trust. See full discussion in **Section 2.3**.

Finding 3

Integration of tool calling within AI agents enable LLMs to autonomously invoke calculators, simulations, and other engineering tools—grounding reasoning in ground truth physics and math. See full discussion in **Section 2.4**.

Several application capabilities have achieved sufficient maturity for near-term deployment.

© 2026 California Institute of Technology. Government sponsorship acknowledged.

Finding 8

AI coding assistants represent the most impactful GenAI capability currently available, with proven functionality ready for mainstream use across engineering disciplines—not merely for software engineers but as cross-cutting tools valuable for scripting, automation, and document preparation. See full discussion in **Section 3.3**.

Finding 9

AI-based Model-Based Systems Engineering (MBSE) platforms exhibit the clearest near-term benefit and maturity among engineering-specific GenAI tools, overcoming the traditional overhead and complexity barriers that limited MBSE adoption historically while enabling new dynamic capabilities beyond classical systems engineering tasks. See full discussion in **Section 3.3**

Finding 12

Systems engineering tasks—requirements derivation, management, and analysis—represent readily achievable GenAI applications, as these language-based activities require contextual understanding across diverse sources, a capability where Large Language Models (LLMs) excel. See full discussion in **Section 4.2**.

However, critical technical gaps constrain GenAI effectiveness for many core engineering tasks.

Finding 10

Limited CAD native integration and spatial reasoning capabilities represent the most significant shortcoming in current LLM applications for mechanical engineering and other disciplines directly tied to 3D geometries, fundamentally constraining the GenAI application space. See full discussion in **Section 3.3**.

Finding 13

GenAI outputs currently require extensive verification due to the mismatch between current system maturity and the high accuracy requirements of engineering work—while RAG systems and agent tool usage provide grounding, verification and validation remains a significant and non-trivial workload. See full discussion in **Section 4.2**.

Beyond technical maturity, organizational factors present equally significant adoption barriers.

© 2026 California Institute of Technology. Government sponsorship acknowledged.

Finding 5

While NASA's institutional knowledge sources—lessons learned, standards, technical documentation—remain largely unintegrated with GenAI systems to date, this presents a high-value opportunity, particularly as RAG-based knowledge access is among the most readily implemented capabilities and NASA's knowledge base is a core organizational strength. See full discussion in **Section 3.2**.

Finding 7

While NASA has initiated a few top-down strategic programs for GenAI in spacecraft design, greater coordinated institutional effort and investment will be required to achieve advanced capabilities such as automated spacecraft system design. See full discussion in **Section 3.2**.

Recommendations

NASA should concentrate investment on capabilities the broader AI ecosystem cannot provide while leveraging commercial foundation models as computational engines. The following five recommendations address workforce development, capability development, and organizational learning; see **Section 5** for a more detailed discussion.

Recommendation 1

Organize a comprehensive, agency-wide AI exploration event delivered as a hybrid program with virtual sessions accessible across all NASA centers supplemented by local in-person activities, providing structured exposure to GenAI capabilities through demonstrations, hands-on workshops, and role-specific breakout sessions.

Context: GenAI adoption varies widely across NASA's workforce, with key barriers including uncertainty about whether tools are relevant to specific job functions, lack of knowledge about effective usage patterns, and insufficient exposure to inspire role-specific applications. An enabled workforce equipped with practical knowledge and inspired by demonstrated applications can drive this technology forward far more effectively than top-down mandates alone. Lawrence Livermore National Laboratory's aiEDGE for Innovation Day provides a proven model, having successfully engaged 3,200 employees across diverse job roles—demonstrating that structured events combining training resources with visible management encouragement create the conditions for workforce-driven innovation.

© 2026 California Institute of Technology. Government sponsorship acknowledged.

Recommendation 2

Develop and publicly host NASA-specific datasets, benchmarks, and challenge problems that address gaps in public resources while meeting agency evaluation needs, engaging the broader research community in developing capabilities serving NASA needs.

Context: Public benchmarks and challenges have proven to be significant contributors to AI progress over the past two decades—ImageNet transformed computer vision research, while engineering challenges like the GE jet engine bracket received 628 entries during the emergence of topology optimization. Applied to NASA-specific technical challenges, this approach could accelerate GenAI capability development in aerospace domains where current foundation models underperform on specialized knowledge and reasoning tasks.

Recommendation 3

Develop and deploy Model Context Protocols (MCPs) providing programmatic access to critical institutional knowledge sources and research literature, enabling both institutional agent platforms and grassroots applications to leverage decades of accumulated technical expertise including NESC products, MAPTIS, NASA STI Repository, and external research databases.

Context: NASA's historical knowledge spanning nearly seven decades remains locked in disconnected systems that GenAI applications cannot easily query. MCPs provide standardized interfaces enabling LLM-based applications to query knowledge sources programmatically. Investment delivers value across all GenAI applications rather than requiring per-application integration efforts, creating multiplier effects as more applications leverage shared knowledge access infrastructure.

Recommendation 4

Mature and expand existing institutional agent platforms (ChatGSFC and JPL's Onyx) to build substantial user adoption and capability breadth across NASA, growing shared agent libraries from dozens to hundreds through active engagement and contribution incentives.

Context: NASA has established foundational agent platform infrastructure, but early adoption metrics reveal significant maturity gaps—Blue Origin's BlueGPT marketplace hosts more than 2,000 deployed agents compared to NASA's tens of agents. The gap reflects limited capability breadth, insufficient user awareness, and absence of sustained organizational encouragement rather than technical infrastructure deficiencies.

© 2026 California Institute of Technology. Government sponsorship acknowledged.

Recommendation 5

Establish a funded position within select flight projects specifically dedicated to identifying, prototyping, and integrating GenAI solutions tailored to mission-specific workflows and challenges, requiring someone familiar with both spacecraft project development and GenAI capabilities funded at minimum half-time but ideally full-time.

Context: Many potentially valuable GenAI applications remain undiscovered because practitioners immersed in immediate project demands lack time to experiment with emerging capabilities, while AI researchers lack the mission context to identify where tools would provide genuine value. A dedicated position addresses this gap by combining spacecraft project knowledge with GenAI expertise in someone whose explicit responsibility is discovering opportunities rather than executing current tasks, providing authentic evaluation within mission contexts where applicability and value become clear through hands-on prototyping.

Recommendation 6

Fund a large-scale demonstration project that applies GenAI-augmented workflows to design and build flight hardware for an actual space mission, substantially exceeding the scope of individual proofs-of-concept to identify organizational factors enabling successful deployment at mission scale.

Context: This demonstration targets the engineering process itself rather than novel hardware capabilities—the objective is to reveal what project-level organization, institutional support structures, process modifications, and cultural changes enable teams to actually realize GenAI's potential at mission scale. Understanding these organizational factors matters more than hardware performance improvements because organizational barriers currently constrain deployment more than technical limitations.

Path Forward

The path forward demands sustained commitment and thoughtful execution toward a transformed aerospace engineering practice. This transformation envisions engineers working in partnership with AI systems that make decades of institutional knowledge conversationally accessible—where lessons from Apollo, Shuttle, and Mars missions inform today's decisions through natural language queries rather than manual searches through disconnected databases. Design exploration becomes dramatically more efficient as GenAI tools handle routine analysis iterations, enabling engineers to evaluate broader solution spaces and focus creative energy on novel approaches rather than repetitive calculations. Verification and documentation workflows that currently consume substantial engineering time become semi-automated, accelerating development cycles without compromising the rigor essential for spaceflight systems. The result is not replacing engineering judgment but amplifying human creativity and expertise with computational capabilities, making the next generation of space missions simultaneously more capable and more achievable within constrained budgets and schedules.

NASA stands uniquely positioned not merely to adopt these tools but to lead the aerospace community in establishing best practices for responsible GenAI deployment in safety-critical engineering. The agency's technical depth spanning seven decades of spaceflight, its role as a trusted authority in the aerospace ecosystem, and its mission-driven culture of innovation create ideal conditions to demonstrate how GenAI

© 2026 California Institute of Technology. Government sponsorship acknowledged.

can enhance rather than replace human expertise. By investing strategically now—building infrastructure, training the workforce, and demonstrating capabilities through focused projects—NASA can establish the standards and practices that guide broader industry adoption, ensuring GenAI deployment advances both technical capability and organizational trust.

© 2026 California Institute of Technology. Government sponsorship acknowledged.

1 Introduction

We are witnessing a transformational transition from deterministic computing—where we tell computers exactly what to do—to generative partnership, where computers propose solutions based on constraints and intent. Traditional computing required precise instructions for every operation. Whether programming a calculation, automating a workflow, or directing a machine tool, humans specified exact procedures and the computer executed them faithfully. Even traditional Artificial Intelligence (AI), while capable of learning patterns from data, operated within narrowly defined tasks requiring specialists to design custom architectures for each application. Generative Artificial Intelligence (GenAI) fundamentally changes this relationship. Through natural language interfaces, these systems enable collaboration rather than mere command execution. Engineers can describe what they need rather than how to achieve it; AI systems propose solutions, generate alternatives, and refine approaches through iterative dialogue. This represents a qualitative shift in human-computer interaction with profound implications across knowledge work—including engineering.

At NASA specifically, GenAI offers a potential pathway to meet mounting pressure for more efficient spacecraft design and development while maintaining technical excellence. The agency faces unprecedented demands to reduce cost and schedule as missions grow increasingly complex with expanding scientific ambitions, yet budgets constrain program scope and workforce capacity. Development cycles measured in years or decades strain relevance in rapidly evolving technological landscapes. Commercial competitors demonstrate faster development cadences, applying modern manufacturing techniques and iterative design approaches that challenge traditional NASA paradigms. GenAI technologies promise to address these pressures through accelerated design iteration, improved access to institutional knowledge spanning decades of missions, reduced documentation burden freeing engineers for creative problem-solving, and leaner teams augmented by AI capabilities covering broader technical scope. The question is not where GenAI might ultimately lead, but what capabilities deliver genuine value today and what strategic investments position NASA to capitalize on emerging opportunities as technology matures.

However, the GenAI landscape is dominated by extraordinary claims that make distinguishing genuine near-term capabilities from aspirational long-term promises difficult. Recently, AI company executives have been predicting the elimination of white-collar jobs within 12-16 months [1]. Software developers publish articles claiming AI coding agents have effectively replaced them in their work [2]. These claims range from plausible projections grounded in demonstrated capabilities to speculative visions requiring breakthroughs that may or may not materialize. For NASA and other engineering organizations considering strategic investments in GenAI, these claims make it challenging to parse the hype from reality:

- What capabilities exist today that can be deployed immediately?
- What developments lie realistically on the near-term horizon?
- Where do fundamental gaps remain that constrain application to NASA workflows?
- What investments—in infrastructure, workforce development, partnerships, and research—position NASA to capitalize on GenAI's transformational potential while avoiding premature commitment to immature technologies?

This NASA Planetary Exploration Science Technology Office (PESTO) study provides systematic assessment of GenAI specifically for spacecraft design and development, delivering both immediate opportunities and a long-term strategic roadmap. We employed a multimethod approach combining expert interviews across NASA centers, aerospace companies, and GenAI vendors; participation in the Text-to-Spaceship Symposium at Goddard Space Flight Center gathering approximately 300 attendees from

© 2026 California Institute of Technology. Government sponsorship acknowledged.

industry, academia, and government laboratories; hands-on pilot demonstrations applying GenAI tools to representative spacecraft design problems; and internal team ideation leveraging aerospace engineering expertise to identify NASA-specific opportunities. Our objectives are to cut through hype and assess what works today, evaluate current GenAI capabilities for aerospace applications, identify NASA-specific opportunities spanning near-term deployment to long-term strategic investment, and provide actionable recommendations grounded in demonstrated capabilities rather than speculative promises.

Our scope focuses specifically on NASA mission Phases A through D: concept and technology development, preliminary design, final design and fabrication, and system assembly, integration, test, and launch [3]. This focus recognizes that Phases A through D represent both a great opportunity and a significant knowledge gap for GenAI deployment within NASA's mission lifecycle. These design-to-implementation phases constitute dominant cost and schedule drivers for NASA missions, with design decisions made during these phases fundamentally determining mission feasibility, development timeline, and program cost before and while hardware is manufactured and deployed. Applying GenAI to accelerate design iteration, improve requirements analysis, enhance knowledge access, and automate documentation directly addresses the cost and schedule pressures that constrain NASA's mission portfolio.

We deliberately exclude Pre-Phase A (concept studies) and Phase E (operations) from this study's scope because these phases already demonstrate clearer paths for GenAI adoption. Pre-Phase A has arguably seen the most GenAI application at NASA to date, as the highly text-based and knowledge-intensive nature of mission formulation translates naturally to current GenAI capabilities. Phase E has historically received the majority of traditional AI investment—computer vision for robotics, autonomous navigation, data science for scientific observations—and GenAI applications to operations appear likely to emerge organically from existing AI expertise in this domain. By contrast, Phases A through D remain relatively unexplored for GenAI deployment despite presenting the greatest potential for reducing mission cost, schedule, and technical risk through improved design workflows and knowledge management.

This report is structured to serve both NASA leadership and technical staff across a spectrum of GenAI expertise, progressing from foundational concepts through evidence to actionable recommendations:

- **Section 2 (Generative AI Technology Landscape)**: Establishes the technical foundation needed to assess generative AI capabilities and limitations for spacecraft applications.
- **Section 3 (Assessment Study)**: Presents findings from industry research and identifies patterns for how GenAI can augment NASA engineering workflows.
- **Section 4 (Pilot Case Study)**: Demonstrates real-world GenAI capabilities and limitations through a hands-on spacecraft design case study.
- **Section 5 (Recommendations and Path Forward)**: Provides actionable recommendations organized by timeframe for deploying GenAI capabilities at NASA.
- **Section 6 (Conclusions)**: Synthesizes key findings and provides final strategic recommendations for NASA leadership.

This report is intentionally comprehensive—GenAI is a nuanced technology domain where confusing and often misleading information pervades broader media coverage, making it essential to provide sufficient context for readers to develop understanding at whatever level of depth serves their decision-making needs. Leadership and decision makers with limited time should prioritize the recommendations (**Section 5**) and conclusions (**Section 6**), while the background (**Section 2**), study approach (**Section 3**), and pilot demonstration (**Section 4**) sections provide technical depth most relevant to engineering staff and practitioners.

© 2026 California Institute of Technology. Government sponsorship acknowledged.

2 GenAI Technology Landscape

Assessing GenAI's potential for NASA requires understanding not just what these systems can do, but how they work and why they fail. The claims surrounding GenAI—from text-to-hardware visions to wholesale replacement of engineering workflows—cannot be evaluated without technical foundation. This section provides that foundation, explaining the core technologies underlying current GenAI systems and their implications for spacecraft engineering applications.

2.1 What Makes This AI Different?

The current wave of GenAI represents a fundamental departure from traditional artificial intelligence approaches. Understanding this distinction is essential for assessing its potential impact on NASA system design and development. Traditional AI systems excelled at specific, narrowly defined tasks but required substantial specialized expertise and custom engineering for each application. Generative AI, by contrast, offers general-purpose capabilities accessible through natural language, fundamentally changing who can use AI and how it can be deployed.

2.1.a Traditional AI: Specialized Tools for Specific Tasks

Traditional machine learning approaches required specialists to design custom model architectures tailored to specific applications with structured inputs and outputs. This development overhead limited broad AI adoption to contexts where the investment justified the expected return. Computer vision systems for object detection and classification, natural language processing for specific text analysis tasks, and specialized predictive models exemplified this paradigm **[4], [5]**. Each application demanded careful feature engineering, model architecture design, training dataset curation, and extensive validation—tasks requiring deep expertise in both machine learning and the target domain.

NASA has successfully deployed traditional AI across multiple mission-critical applications. Mars rover navigation systems employ computer vision for autonomous terrain assessment and path planning **[6], [7]**. Mission operations leverage predictive models for anomaly detection in spacecraft telemetry and resource optimization **[8], [9], [10]**. Science missions utilize machine learning for data analysis, pattern recognition, and predictive modeling **[11], [12], [13], [14], [15]**. While these applications demonstrate AI's value, each required significant custom development by specialized teams.

2.1.b The Generative AI Paradigm Shift

GenAI fundamentally differs from traditional predictive models in both capability and accessibility. *Predictive (discriminative) models* classify inputs or make predictions based on patterns learned from training data (ex. detect obstacles or points of interest in an image, identify telemetry anomalies). *Generative models*, by contrast, create new content—generating text, images, code, or other outputs based on patterns learned during training **[16]**.

The breakthrough enabling broad GenAI adoption is the natural language interface. LLMs accept instructions and queries in plain language, eliminating the need for users to possess machine learning expertise or programming skills for many tasks. This accessibility represents a profound departure from traditional AI, where deploying even a pre-trained model typically required technical integration work. Equally important, natural language interfaces enable systems to operate in ambiguous problem spaces where conventional conditional computing struggles. Traditional programs require explicit handling of all possible outcomes and cases, demanding precise specifications upfront. Natural language allows engineers to work dynamically past these bounds—describing intent and constraints rather than enumerating every condition—while the AI interprets ambiguous requirements and proposes solutions. This flexibility lowers

© 2026 California Institute of Technology. Government sponsorship acknowledged.

the barrier to entry, enabling domain experts in engineering, science, and management to interact directly with AI systems without intermediary technical specialists.

> Finding 1
>
> GenAI systems can operate under ambiguity, accepting intent and constraints rather than requiring exhaustive specifications—a fundamental shift from traditional conditional computing.

2.2 Large Language Models: Capabilities and Limitations

Large Language Models (LLMs) form the foundation of current GenAI systems. LLMs demonstrate impressive capabilities in language understanding, reasoning, and content generation, yet they also exhibit fundamental limitations that constrain their reliability and applicability. Understanding how these models work, what they can accomplish, and where they fail is essential for deploying them effectively in engineering applications.

2.2.a How LLMs Work: A High-Level View

At their core, LLMs are sophisticated autocomplete systems that predict the next most probable token given preceding text (**Figure 1**). Specifically, text is broken into *tokens*—sub-word units that efficiently encode vocabulary—where a word like "spacecraft" might be one token, while "uncharacteristic" might be split into "un-", "character", "-istic." The model doesn't "think" or "understand" in any human sense; instead, it calculates probability distributions based on statistical patterns learned from training data. This is fundamentally different from rule-based systems or knowledge databases—LLMs perform pattern matching at massive scale, predicting likely continuations rather than retrieving stored information or applying programmed logic. Running this prediction process recursively—feeding each newly generated token back as input—enables models to generate complete sentences and paragraphs that maintain coherence across extended text.

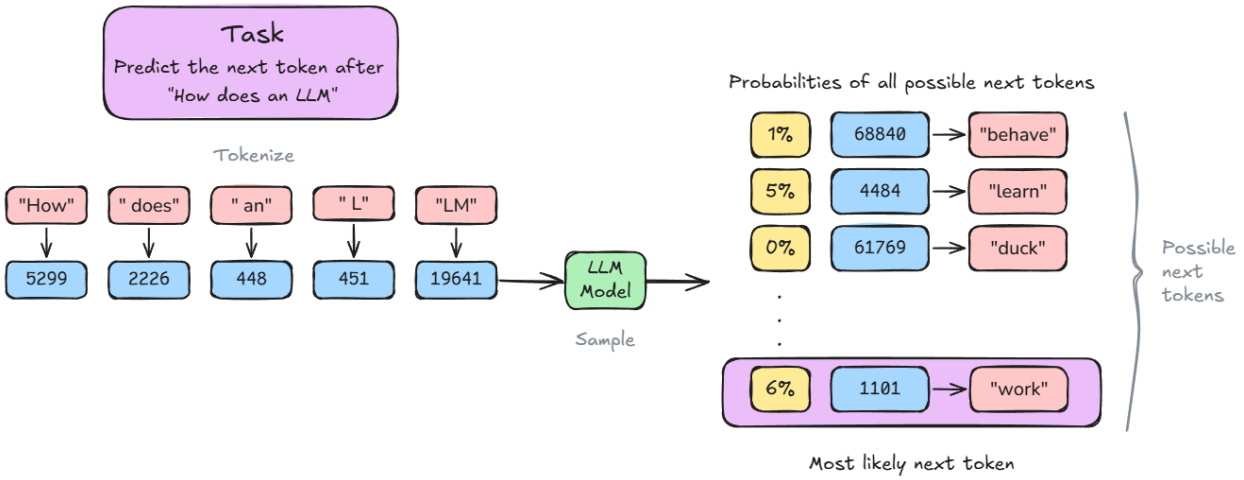


Figure 1: LLMs generate text by recursively predicting the next most probable token given preceding text. Given the input "How does an LLM", the text is first tokenized, then the model calculates probabilities for all possible next tokens and selects the highest probability option—in this example, the word "work". Running this prediction process recursively generates complete sentences and paragraphs.

© 2026 California Institute of Technology. Government sponsorship acknowledged.

What makes this pattern matching so effective is the underlying transformer model architecture—the "T" in GPT (Generative Pre-trained Transformer)—which enables models to capture long-range understanding and context across long strings of text [16]. Older language models processed text sequentially from left to right, often "forgetting" earlier context by the time they reached the end of a long sentence or paragraph. Transformers broke this limitation through *self-attention mechanisms* that analyze all tokens simultaneously, weighing the relevance of every word to every other word [17]. Consider the phrase "The spacecraft's propulsion system failed during orbit insertion, which the flight software detected and compensated for by switching to the backup thrusters." Self-attention enables the model to connect "which" to the entire failure event rather than just the nearest noun ("orbit insertion"), and to understand that "compensated for" refers back to this failure despite being separated by intervening words—tracking causal relationships across the sentence beyond simple adjacent-word patterns. This architecture scales efficiently to billions of parameters, enabling massive models that capture nuanced language patterns and complex reasoning chains.

Despite these impressive scaling properties, the *context-window* of the transformer attention mechanism is ultimately limited in size. Because computational demands scale with the depth of self-attention mechanisms—every token must attend to every other token—context windows require an explicit cutoff beyond which processing becomes prohibitively expensive or ineffective. Current systems range from thousands up to a million tokens, but even the largest windows are often insufficient to capture the full scope of complex tasks and large information sources. This means that long-term memory, project-specific knowledge accumulated over weeks or months, and understanding that spans massive documentation sets all require additional infrastructure beyond the LLM itself.

To produce well behaved models with comprehension-like behavior, modern LLMs undergo a multi-stage training process. *Unsupervised pre-training* forms the foundation (the "P" in GPT): models "read" hundreds of billions of phrases spanning books, articles, code repositories, and web content, learning statistical patterns without explicit instruction. This is the "reading the internet" phase where general language understanding emerges—the model absorbs not just vocabulary and grammar, but facts, reasoning patterns, and domain knowledge encoded in the training corpus. Following pre-training, *supervised fine-tuning* adapts the model to follow instructions, answer questions helpfully, and maintain coherent conversations by training on curated examples demonstrating desired behaviors. This step produces the conversational characteristics users experience in modern AI assistants: polite, helpful responses; acknowledgment of limitations; refusal of harmful requests; and the reassuring demeanor that makes these systems feel like collaborative partners rather than unpredictable text generators. After *supervised fine-tuning*, additional fine-tuning such as Reinforcement Learning from Human Feedback (RLHF) can improve reasoning capabilities, though approaches remain less standardized and constitute an active area of research.

While initial LLMs processed only text, newer multimodal LLMs are capable of directly operating on additional data formats like images, audio, and video [16]. Like text, these data modalities are tokenized and processed by the LLM using the same self-attention mechanisms—the model learns statistical patterns relating data features to language concepts. This remains a highly active area of research, with startup companies emerging to develop additional data modalities. For engineering applications, researchers are exploring how Computer-Aided Design (CAD) geometry models, simulation meshes, simulation results, and technical drawings could allow LLMs to reason directly about physical designs rather than relying solely on textual descriptions that lose essential geometric and topological information.

© 2026 California Institute of Technology. Government sponsorship acknowledged.

2.2.b Core Capabilities

Natural language understanding and generation enables these systems to comprehend technical documentation, answer questions, generate reports, and explain complex concepts in accessible language. Models parse intricate sentence structures, resolve ambiguous references, and maintain thematic coherence across extended passages.

- *Reasoning and problem-solving within context* represents a particularly valuable capability. LLMs can break down complex problems into steps, apply logical reasoning chains, and synthesize information from multiple sources within their context window. While this reasoning operates through pattern matching rather than formal logic, it produces useful results for many practical tasks. It should be noted that the reasoning steps LLMs articulate should not be relied upon as accurate reflections of their internal decision-making processes—reinforcement learning optimizes models to generate these steps because they improve output quality, not because they reveal genuine reasoning chains **[18]**.
- *Code generation and analysis* has proven especially effective. LLMs trained on code repositories can generate functions from natural language descriptions, debug existing code, explain program behavior, convert between programming languages, and suggest optimizations. For NASA tasks involving software development, testing, or data analysis, these capabilities offer substantial productivity gains.
- *Information synthesis from unstructured sources* enables LLMs to extract relevant information from technical documents, summarize lengthy reports, identify patterns across multiple sources, and present findings in structured formats. This capability proves particularly valuable for accessing historical mission data, requirements documents, and technical specifications—exactly the unstructured knowledge bases NASA possesses in abundance.
- *Translation between representations*—converting natural language to code, structured data to prose, mathematical notation to explanations, and vice versa—allows LLMs to serve as interfaces between different technical domains. An engineer could describe a data analysis task in plain language and receive executable code, or query a database using natural language rather than SQL.

2.2.c Fundamental Limitations

Knowledge cutoff dates mean models possess information only from their training data, typically ending months or years before deployment. LLMs cannot access real-time information, retrieve current documentation, or learn from interactions without retraining or augmentation with external knowledge sources or tools.

Hallucinations and factual errors present a critical reliability concern. LLMs generate text that sounds plausible but may contain fabricated information, incorrect specifications, or nonexistent references. This occurs because models optimize for generating coherent, contextually appropriate text rather than verifiable truth. For applications where accuracy is critical, additional infrastructure such as Retrieval Augmented Generation (**Section 2.3**) and tool use (**Section 2.4**) can better ground model outputs, though comprehensive verification and validation remain essential.

The *lack of true understanding* distinguishes LLMs from human reasoning. These systems predict likely next words based on statistical patterns rather than comprehending meaning or applying causal reasoning. While this approach produces impressively capable behavior, it fails unpredictably on tasks requiring genuine comprehension, physical intuition, or novel problem-solving. LLMs can describe engineering principles they don't truly understand, leading to subtle errors in application. Notably, LLMs struggle

© 2026 California Institute of Technology. Government sponsorship acknowledged.

with numerical calculations—even basic arithmetic—because they predict tokens rather than computing values, making them unreliable for engineering calculations without external computational tools.

Limited training modalities constrain LLM capabilities for engineering applications. Current foundation models are trained primarily on text and images, data formats that capture only a limited scope of engineering work. Spatial reasoning, three-dimensional geometry, and physics-based simulation—fundamental to engineering—are not natively supported by models trained on language and two-dimensional visual data. Language proves a poor approximator for geometric and spatial concepts: describing complex geometries, spatial relationships, interference conditions, or kinematic constraints in text loses the precision and intuition that native geometric representations provide. Even when augmented with tool-calling harnesses that connect LLMs to CAD systems or simulation engines, models lack fundamental understanding of how geometry interacts with itself and the physical world—they can orchestrate tool calls based on linguistic patterns but cannot reason about geometric properties, spatial constraints, or physical behavior in the way engineers manipulate and understand these concepts through native geometric representations. This limitation creates fundamental gaps for applications requiring deep geometric reasoning, constraining autonomous design capabilities until models either incorporate native geometric representations or achieve sufficient sophistication to overcome the language-geometry representation gap.

Context window constraints limit how much information models can consider simultaneously. Even with context windows spanning hundreds of thousands of tokens, complex systems with extensive documentation, requirements, and design history may exceed this capacity. Models cannot maintain coherent reasoning across information exceeding their context limit without external memory systems. Additionally, LLMs lack persistent memory—each conversation starts fresh unless external systems store and retrieve prior interactions. Emerging techniques like prompt compaction and intelligent context pruning show promise for managing these limitations, but remain immature with no consensus approaches yet established.

These limitations do not negate LLM utility but rather define boundaries for responsible deployment. Effective applications will combine LLM strengths with complementary systems—retrieval augmentation for knowledge access, verification workflows for accuracy, domain-specific tools for specialized reasoning, and human oversight for critical decisions. Understanding both capabilities and limitations enables realistic assessment of where LLMs can provide value and where alternative approaches remain necessary.

2.3 Retrieval Augmented Generation: Connecting LLMs to Knowledge

Retrieval Augmented Generation (RAG) addresses some of the most critical LLM limitations—knowledge cutoff dates, hallucinations, and lack of traceability—by grounding model responses in actual source documents. RAG operates as an application layer built on top of foundation LLMs, not modifying the model's training state but rather implementing sophisticated prompt engineering that augments user queries with relevant information retrieved from document collections or knowledge databases. When a user poses a query, the RAG system first searches for relevant passages, then provides these retrieved passages as additional context to the LLM alongside the original question—the LLM itself remains unchanged, but receives enriched input that grounds its response in actual source material rather than relying solely on training data patterns. For NASA applications, where decades of technical documentation, mission reports, standards, and lessons learned exist in largely unstructured formats, RAG represents a transformational capability. It enables natural language access to institutional knowledge while providing the citations and traceability essential for verification and validation.

© 2026 California Institute of Technology. Government sponsorship acknowledged.

2.3.a How RAG Works

RAG is an application layer on top of an LLM that operates through a two-stage pipeline (**Figure 2**): document preparation (performed initially and reindexed as the knowledge base updates) and query-time retrieval and generation (performed for each user query) **[19]**. Understanding this workflow clarifies both RAG's capabilities and its implementation requirements.

Document preparation converts source documents into semantically searchable units through a multi-step process. The system ingests documents (PDFs, Word files, presentations, technical drawings), divides them into chunks of a few hundred to a few thousand characters with preserved metadata for citation, then transforms these text chunks into numerical vector representations called embeddings that capture semantic meaning. Semantically similar text produces similar vectors—"spacecraft thermal control system" and "satellite temperature regulation" yield nearby vectors despite different wording—enabling the system to find relevant documents even when terminology differs from the query. These embeddings are stored in specialized vector databases optimized for fast similarity search across millions of document chunks. However, handling complex document elements—tables, figures, equations, and technical drawings—remains an active area of development where implementations vary significantly in effectiveness, with no consensus approach for preserving semantic meaning from these non-textual information sources.

Query-time retrieval and generation occurs in two steps when a user submits a query. First, the system converts the query into an embedding, searches the vector database for the most similar chunk embeddings, and retrieves the corresponding text chunks (typically 5-20 chunks, balancing comprehensive context against LLM context window limits). Second, the retrieved chunks are injected into the LLM's context window before the user query—a process that typically happens transparently in the background without visibility to the user—and the LLM generates a response grounded in this augmented context, producing outputs with citations to specific source documents.

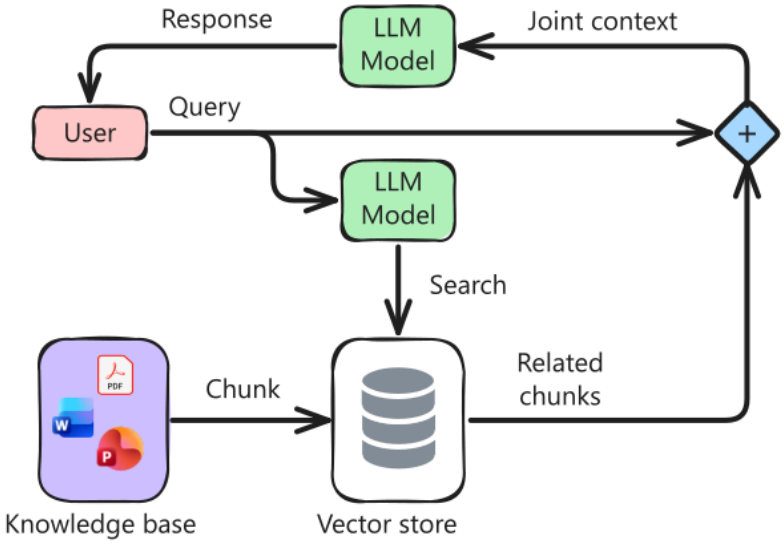


Figure 2: Retrieval-augmented generation operates through a two-stage pipeline. During offline document preparation, source documents are ingested, divided into semantic chunks, converted into vector embeddings that capture meaning, and stored in a vector database. At query time, the user's question is embedded and matched against the database to retrieve relevant chunks, which are then provided alongside the query to the language model to generate a grounded response with citations.

© 2026 California Institute of Technology. Government sponsorship acknowledged.

2.3.b Key Benefits for NASA and General Engineering Applications

Access to mission-specific documentation without retraining models allows organizations to deploy general-purpose LLMs while still providing access to proprietary technical knowledge. Rather than fine-tuning models on institutional documentation—a process requiring substantial computational resources—RAG systems simply add documents to the retrieval database. Engineers can query Mars rover thermal designs, Europa Clipper power systems, or Gateway module specifications using the same underlying LLM, with retrieval selecting relevant mission documentation.

Reduced hallucinations through grounding in source material addresses the most critical reliability concern for NASA applications. By constraining LLM responses to information present in retrieved documents, RAG systems dramatically reduce fabricated specifications, nonexistent references, and incorrect technical claims. While verification remains necessary, responses grounded in authoritative source documents prove far more reliable than unconstrained generation.

Citations and traceability to original documents represent an essential capability for verification and validation. RAG systems can identify which source documents supported each claim in a generated response, enabling engineers to verify critical information against authoritative sources. This traceability proves particularly important for NASA applications, where incorrect specifications or design parameters can have serious consequences.

Keeping knowledge current as new documents are added allows RAG systems to remain up-to-date without model retraining. When NASA publishes updated standards, completes new missions, or generates additional lessons learned, engineers can simply add these documents to the RAG system's database. Subsequent queries will retrieve and incorporate this new information. This capability contrasts sharply with LLMs' static knowledge cutoffs.

Querying natural language across heterogeneous document types enables engineers to search across PDFs, Word documents, technical reports, presentations, and potentially even metadata from technical drawings using simple natural language queries. Rather than remembering which SharePoint site holds thermal design documentation or which database contains test results, engineers can ask questions in plain language and allow the RAG system to search across all available sources.

Finding 2

RAG systems are essential for GenAI in engineering applications, not only grounding generated content in authoritative knowledge bases but also providing traceability to the source material—a critical component for establishing trust.

2.3.c Limitations and Challenges

Context window limits constrain how many retrieved chunks can inform the LLM's response. Even with context windows spanning hundreds of thousands of tokens, systems can include only a limited number of retrieved chunks. If relevant information spans many documents, or if the retrieval process ranks critical information below the cutoff, the LLM lacks necessary context for accurate responses.

Document preprocessing presents particular challenges for technical documentation. PDFs with complex layouts, multi-column text, embedded tables and figures, mathematical notation, and technical diagrams require careful processing to extract meaningful text chunks. Tables and figures may contain critical information that pure text extraction misses. Maintaining document structure—understanding that

© 2026 California Institute of Technology. Government sponsorship acknowledged.

certain paragraphs belong to specific subsections or relate to particular figures—requires sophisticated preprocessing. This remains an active area of development, with commercial and open-source tools demonstrating widely varying effectiveness in handling complex technical documents.

Versioning and updates become complex when documents evolve. If the RAG system contains multiple versions of a requirements document or design specification, retrieval may return outdated information without clearly indicating version numbers. Keeping vector database embeddings synchronized with source documents as those documents are updated requires careful engineering. NASA applications, where version control and configuration management are critical, require RAG systems to track document versions and clearly indicate which version informed each response.

Computational costs include both storage for large vector databases and compute resources for embedding generation. Embedding millions of pages of technical documentation requires substantial initial computation. Vector databases storing high-dimensional embeddings for these documents consume significant storage. Query-time retrieval and LLM generation require additional compute. While less expensive than training custom models, RAG deployment still requires infrastructure investment.

Security and access control must ensure users can only retrieve documents they have permission to access. NASA handles various classification levels and export-controlled technical information. RAG systems must integrate with existing access control systems to prevent unauthorized document access, while still enabling efficient search across documents a user can legitimately access. This requires careful security engineering.

Despite these challenges, RAG represents a mature, deployable technology already proving valuable in aerospace contexts. Understanding its limitations enables realistic deployment planning while recognizing its substantial advantages for accessing institutional technical knowledge.

2.4 Beyond Chat: Agentic Frameworks

LLMs initially appeared as conversational interfaces—chatbots that answered questions and generated text based on prompts. While useful, this paradigm limited LLMs to passive information providers requiring humans to orchestrate all actions. The emergence of *agentic frameworks* fundamentally expands LLM capabilities by enabling these systems to use tools, execute actions, plan complex tasks, and operate with increasing autonomy. Rather than simply responding to queries, AI agents can retrieve information from databases, execute code, analyze data, interact with external systems, and iteratively refine solutions. This transformation from passive chat to active problem-solving agents opens substantially broader application opportunities for engineering.

2.4.a Tool Use and Function Calling

The natural language capabilities that enable LLMs to understand conversational queries also enable them to interact with general-purpose tools and application programming interfaces. *Tool use*, sometimes called *function calling*, allows LLMs to recognize when a user query requires external actions—searching a database, performing calculations, executing code, retrieving files, or calling specialized software—and to invoke appropriate tools with correctly formatted parameters. This capability resembles automated scripting, with the critical distinction that the LLM dynamically determines which tools to invoke and what parameters to provide based on conversational context, rather than following pre-programmed sequences.

© 2026 California Institute of Technology. Government sponsorship acknowledged.

At a high level, the system architecture for tool use follows a cycle: the user submits a query, the LLM determines what tools (if any) are needed and generates structured function calls, the system executes those function calls using external tools or APIs, results return to the LLM as additional context, and the LLM synthesizes results into a natural language response. This enables LLMs to access current information beyond their training data, perform precise calculations, execute data analyses, and interact with existing engineering software ecosystems. Various architectural approaches to agents exist, with a common framework being the ReAct (Reason+Act) methodology [20], which structures this cycle as an iterative loop of Thought (reasoning about the problem), Act (executing a tool or action), and Observe (processing the results) (**Figure 3**).

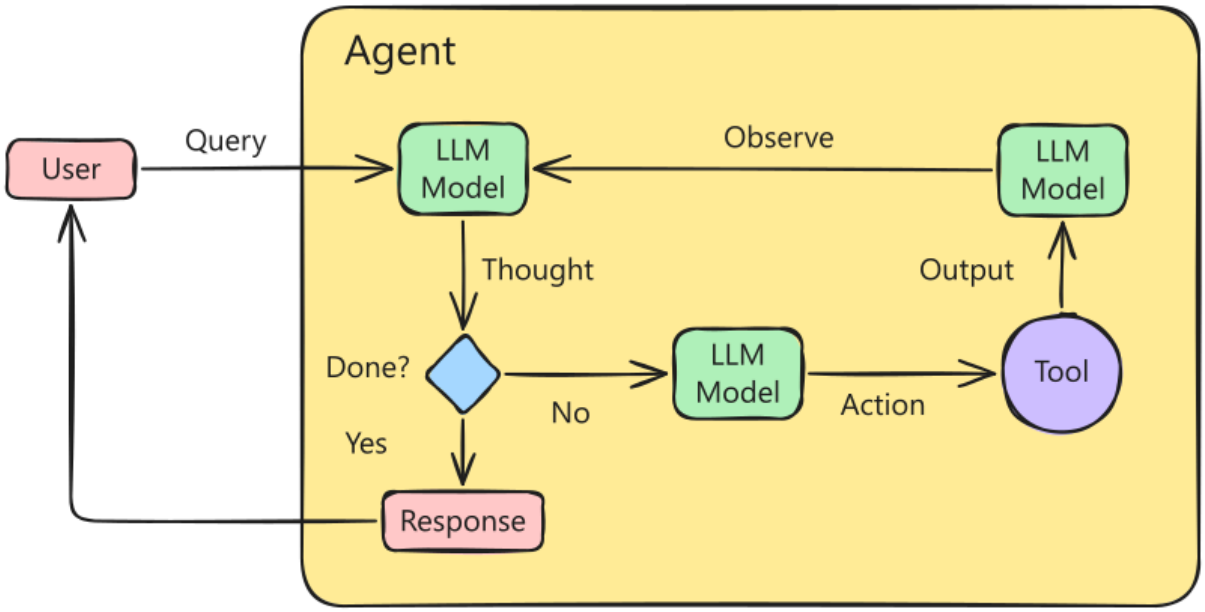


Figure 3: ReAct agent framework showing the iterative cycle of Thought, Act, and Observe. Given a user query, the agent enters the Thought phase where it reasons about what action to take, proceeds to the Act phase where it generates a structured tool call and executes it using external APIs or software, then moves to the Observe phase where results are returned as context. This Thought-Act-Observe cycle repeats iteratively as the agent gathers information and refines its understanding before synthesizing a final answer.

For engineering applications, tool use enables LLM integration to ground reasoning and decision making with physical reality. Rather than replacing specialized engineering tools, AI agents can orchestrate these tools—querying requirements databases, running analyses in MATLAB or Python, retrieving CAD model parameters, accessing configuration management systems, and synthesizing results across multiple tools.

Standardizing these tool connections presents a significant challenge as each data source, API, and software system traditionally requires custom integration code. The *Model Context Protocol* (MCP) addresses this by providing a universal, open-source standard for connecting LLMs to diverse data sources and tools [21]. Rather than implementing bespoke integrations for each database, file system, or API, developers can create MCP servers that expose functionality through a standardized interface. Any MCP-compatible AI system can then connect to these servers without additional custom code.

© 2026 California Institute of Technology. Government sponsorship acknowledged.

Finding 3

Integration of tool calling within AI agents is essential for engineering applications, enabling LLMs to autonomously invoke calculators, simulations, and other engineering tools—grounding reasoning in ground truth physics and math.

2.4.b Autonomous Agents

Autonomous agents extend tool use with planning, self-correction, and iterative refinement capabilities, enabling them to decompose complex tasks into subtasks, execute multi-step workflows, evaluate intermediate results, and adjust strategies when initial approaches fail. Given a high-level objective such as "analyze thermal margins for all spacecraft components given updated mission profiles," an agent can decompose this into sequential steps (retrieve thermal models, obtain mission profiles, execute analyses, identify insufficient margins, generate report), recognize dependencies between subtasks, and execute the workflow systematically. When tool calls fail or initial results prove insufficient, agents can diagnose errors, adjust their approach, and retry with corrections—a capability particularly valuable for engineering tasks where initial approaches frequently require refinement.

Autonomous agentic tools are becoming more commonplace in commercially available tools. Examples include,

- Deep Research agents, such as those found in chat interfaces like Google's Gemini, OpenAI's ChatGPT, and Anthropic's Claude, combine LLM reasoning with web search tools to conduct comprehensive research on complex topics, iteratively formulating search queries, analyzing results, and identifying knowledge gaps requiring additional investigation.
- Claude Code provides LLMs with tools for reading files, executing bash commands, and editing/executing code, along with planning capabilities to iteratively write, test, and debug code.
- LangChain provides a comprehensive framework for building custom agents with configurable tool sets, planning strategies, and memory implementations.

2.4.c Multi-Agent Systems

Multiagent systems coordinate multiple specialized autonomous agents to tackle problems beyond single-agent capabilities, enabling parallel execution, domain specialization, and sophisticated collaboration patterns [22]. Rather than deploying one generalist agent, multiagent architectures assign specialized agents to different aspects of complex tasks—in aerospace design workflows, for example, separate agents might handle requirements analysis, geometric design, structural analysis, and documentation generation, coordinating by sharing intermediate results and collectively converging on solutions. These agents coordinate through orchestration patterns including sequential routing (tasks flow through agents in predefined order), hierarchical delegation (manager agent coordinates specialist agents), or collaborative negotiation (agents jointly allocate tasks and refine solutions). Multiagent systems offer several advantages: parallel execution reduces overall workflow time, modularity enables updating individual agents without system redesign, fault isolation contains failures to specific agents, and distributed expertise allows incorporating domain-specific models as specialized agents.

Potential NASA applications for multiagent systems include: integrated design environments where specialized agents handle different design aspects (geometry, thermal, structures, power, communication) while coordinating to maintain system-level consistency (**Figure 4**); automated review processes

© 2026 California Institute of Technology. Government sponsorship acknowledged.

with agents specializing in requirements compliance checking, design standard verification, analysis validation, and documentation completeness; and mission planning systems where agents specialize in trajectory design, resource allocation, risk assessment, and operations scheduling while collaborating to produce integrated mission plans.

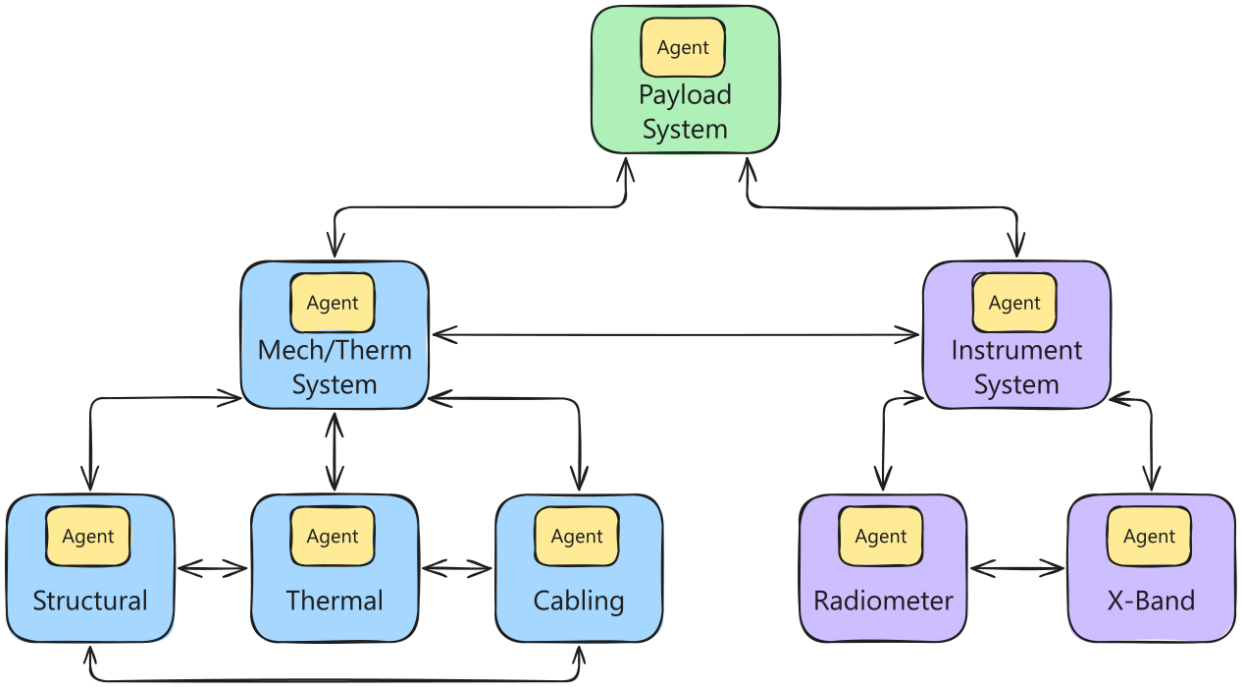


Figure 4: Concept level multiagent system architecture for spacecraft payload design showing hierarchical coordination of specialized agents. Each agent focuses on a specific domain (requirements, structures, thermal) and communicates through various levels of the hierarchy.

Current multiagent systems remain largely experimental, with most production deployments employing relatively simple architectures. However, the technology trajectory clearly points toward increasingly sophisticated agent collaboration. As individual agent capabilities mature and orchestration frameworks develop, multiagent systems will likely become practical for complex engineering workflows requiring coordinated expertise across multiple domains.

© 2026 California Institute of Technology. Government sponsorship acknowledged.

3 Assessment Study

GenAI capabilities for engineering applications remain obscured by vendor marketing claims and technology hype, making it difficult to distinguish genuine capability from aspirational vision. This study aims to cut through this hype and assess both the current state and realistic potential of GenAI in the context of spacecraft design and implementation. To accomplish this, the study examines three interconnected dimensions:

- **Aerospace industry practice**: Assess implementation approaches that organizations have adopted, enabling factors that facilitate successful deployment, and barriers that constrain broader adoption
- **Software ecosystem maturity**: Characterize the rapidly evolving GenAI software ecosystem and evaluate its readiness for aerospace applications
- **Application space**: Characterize potential use cases for GenAI in aerospace engineering, evaluating technology readiness, required scope of investment, and potential return on investment across different applications

3.1 Study Design

3.1.a Industry Engagement

We engaged directly with aerospace practitioners and GenAI ecosystem participants to gain an in-depth understanding of organizational experiences and technical challenges. We conducted conversational, exploratory interviews with practitioners across NASA centers, aerospace companies, national laboratories, and GenAI vendors. The open-ended interview format accommodated the study's broad scope and allowed participants to highlight issues most relevant to their organizations. These interviews included both scheduled meetings and impromptu interactions that emerged organically through symposium participation and collaborative discussions.

We also attended the inaugural Text-to-Spaceship Symposium put on and hosted by Goddard Space Flight Center in January 2026. This one day event brought together approximately 300 participants from NASA centers, industry, academia, and government laboratories. The symposium provided direct exposure to emerging implementations through technical demonstrations, presentations of early adoption experiences, and extensive discussions of both opportunities and challenges practitioners encountered in real deployment scenarios.

3.1.b Research and Analysis Methods

Beyond direct engagement with practitioners, the study team conducted internal research and analysis. This included literature review of technical publications, vendor materials, and industry case studies documenting GenAI applications. We performed software ecosystem analysis examining foundation models, development frameworks, and application categories to understand the rapidly evolving tool landscape. Internal team ideation sessions leveraged aerospace engineering expertise to explore potential applications and identify technical requirements specific to NASA workflows.

The study itself employed GenAI tools as research assistants, demonstrating the technology's current capabilities in practice. This human-AI collaborative investigation served dual purposes: accelerating the research process while simultaneously providing firsthand insight into effective collaboration patterns for knowledge work. Human researchers led all reasoning, decision-making, and validation activities. We employed traditional research methods including conducting expert interviews, reviewing technical literature, and facilitating symposium discussions. GenAI augmented this work through transcription of

© 2026 California Institute of Technology. Government sponsorship acknowledged.

interviews and presentations, summarization of lengthy documents and discussion sessions, literature synthesis across diverse sources, and pattern identification across large information sets. This methodological approach both enabled more comprehensive analysis within project constraints and yielded practical insights into productive human-AI collaboration that inform our recommendations.

3.1.c Study Participants and Scope

This study engaged participants from across the aerospace and GenAI ecosystems between September 2025 and April 2026. The study timeline is significant given the rapid pace of GenAI evolution—capabilities, tools, and deployment patterns changed measurably even within this six-month window. Our observations and assertions account for this temporal context; findings reflect the state of technology and practice as of early 2026.

Aerospace Ecosystem Participants:

- **NASA Centers**
  - Jet Propulsion Laboratory
  - Goddard Space Flight Center
  - Marshall Space Flight Center
  - Ames Research Center
  - Langley Research Center
  - Johnson Space Center
- **Industry**
  - iMETALX
  - Motiv Space Systems
  - Xometry
- **National Laboratories**
  - Lawrence Livermore National Laboratory

Software Ecosystem Participants:

- **AI-Native Companies** (built from inception with GenAI capabilities)
  - Celedon (DaVinci Model-Based Systems Engineering (MBSE) platform)
  - P1 AI
  - Mecado
  - Pensa Technologies
  - Zoo
- **Incumbent Software Companies** (integrating GenAI into existing platforms)
  - Autodesk (CAD/CAE)
  - InfinitForm
  - nTop
  - Synera
  - XMPro
  - NVIDIA

© 2026 California Institute of Technology. Government sponsorship acknowledged.

- ANSYS

3.2 State of Practice in Aerospace

Our assessment revealed that beyond technical capabilities and maturity considerations, organizational factors present substantial barriers to near-term GenAI adoption in aerospace. While the technology demonstrates clear value in multiple application domains, and grassroots innovation shows promising early results, systemic factors constrain broader deployment. These organizational factors span four categories: infrastructure, people, institutional culture, and strategy.

3.2.a Infrastructure

Data Sensitivity and Secure Access

Secure GenAI tools capable of handling varying levels of sensitive aerospace data now exist, but deployment remains uneven across organizations, and varied even within an organization. A few examples include,

- NASA has deployed an array of agency wide capabilities, including a custom deployment called ChatGFSC (International Traffic in Arms Regulations (ITAR), Export Administration Regulations (EAR), Controlled Unclassified Information (CUI)) along with commercial tools such as Google Gemini for Workspace (CUI), Microsoft 365 Copilot (ITAR, EAR, CUI), Amazon Bedrock (ITAR, EAR, CUI), and Azure AI (ITAR, EAR, CUI)
- JPL has deployed an array of internal capabilities and services, including a custom deployment called ChatHPC (ITAR, EAR, CUI), Google Gemini for Workspace (CUI), Microsoft 365 Copilot (ITAR, EAR, CUI), Amazon Bedrock (ITAR, EAR, CUI), and betas of specialized services such as Onyx, Windsurf, and Claude Code.
- Blue Origin has a diverse set of tools supported by AWS, including a secure LLM gateway, an agent market place, and multi-agent orchestration platform [23].
- Smaller aerospace companies are tending more towards as-needed individual access that are often unable to handle sensitive data.

The array of tools available within a single organization speaks to the specialized capabilities of various tools (no one toolset meets all needs) and the rapidly changing landscape. This is also particularly true for government cloud solutions, which often lag behind commercial state-of-the-art due to additional security validation requirements, creating tension between security needs and access to the latest capabilities.

Finding 4

NASA's provisioning of GenAI tools spanning multiple data sensitivity levels (including ITAR and EAR data) has been effective and essential to the current degree of workforce adoption and organizational success with these technologies.

Knowledge Source Integration and Permission Management

Organizations are currently struggling to roll out institutional knowledge systems. Decades of data stored across heterogeneous legacy systems present significant integration challenge, many of which lack well-defined APIs or MCP implementations required for automated ingestion. Large organizations must also

© 2026 California Institute of Technology. Government sponsorship acknowledged.

manage information silos where data is segregated by user privilege level, though RAG systems can accommodate this through user-specific knowledge access controls. Institutions are working toward broader automated integration, but rollout timelines depend heavily on knowledge source extent and institutional complexity. With the exceptions of Blue Origin's BlueGPT ecosystem [23] and a beta implementation of Onyx at JPL, most knowledge-based systems are currently being build from the ground up using isolated databases for specific tasks.

Finding 5

While NASA's institutional knowledge sources—lessons learned, standards, technical documentation—remain largely unintegrated with GenAI systems to date, this presents a high-value opportunity, particularly as RAG-based knowledge access is among the most readily implemented capabilities and NASA's knowledge base is a core organizational strength.

Agent and Capability Sharing Platforms

Most organizations lack infrastructure for sharing GenAI capabilities across teams, forcing individuals to rebuild similar tools independently and limiting knowledge transfer. Grassroots tools developed by individuals remain difficult for others to replicate and deploy—at best, developers share code via GitHub, which proves inaccessible for most potential users without programming expertise. Organizations need searchable platforms with discovery mechanisms where users can find, deploy, and adapt capabilities others have created. Blue Origin's BlueGPT platform demonstrates what such infrastructure enables: an AWS-based agent marketplace hosting more than 2,000 deployed agents across the company [23]. NASA's ChatGSFC Agent Marketplace and JPL's Onyx platform (currently in beta) follow a similar approach but remain in earlier stages of deployment with smaller agent bases. Key features for effective sharing include download metrics to identify valuable tools, version control for tracking improvements, and simplified setup processes that reduce deployment friction. Maintaining security and data sensitivity controls across shared capabilities remains the primary technical challenge.

Data Longevity Concerns

Organizations face vendor lock-in risks analogous to CAD or PLM system dependencies—substantial investment in one platform makes migration difficult or impractical. Uncertainty exists about long-term vendor viability, raising questions about data accessibility if vendors cease operations or discontinue products. These concerns affect strategic planning and investment decisions but have not yet crystallized into concrete mitigation strategies across the industry.

3.2.b People

Early Adopters and Grassroots Innovation

Most GenAI work in aerospace currently occurs at the grassroots level through bottom-up experimentation and custom tool development. This pattern represents technology pull—end users solving problems they encounter daily—rather than technology push from management directives. Individual champions drive adoption within their organizations, building tools that deliver immediate value and demonstrating capabilities to colleagues. This grassroots activity proves valuable for organizational learning and experience building; however, faces scalability limitations for more sophisticated applications and deployments. Without organizational sharing mechanisms, teams often rebuild similar solutions independently, duplicating effort and fragmenting knowledge about what works.

© 2026 California Institute of Technology. Government sponsorship acknowledged.

Finding 6

The majority of GenAI applications at NASA currently emerge from grassroots innovation rather than top-down institutional strategy.

Individual Resistance and Concerns

A small segment of the workforce remains ideologically or personally opposed to using GenAI tools. This anti-AI sentiment reflects varying rationales including concerns about job displacement, ethical objections to AI development, or philosophical opposition to the technology. As such, some individuals express concern about potential requirements to use AI tools.

Poor experiences with GenAI tools—whether direct or observational—also create disproportionate negative perceptions that discourage further exploration. Direct negative experiences occur when individuals attempt to use GenAI for inappropriate tasks beyond current capabilities or without adequate understanding of effective prompting techniques, leading to frustration and dismissal of the entire technology rather than recognizing task-specific limitations. Observational negative experiences prove equally damaging: exposure to poorly generated content from others—obviously AI-written emails with awkward phrasing, hallucinated technical claims in reports, or low-quality outputs lacking human review—creates skepticism about the technology's utility even among those who haven't used it themselves. These experiences compound to form lasting impressions that "GenAI doesn't work," when the reality is often misapplication rather than fundamental capability failure. This bias presents particular challenges because negative experiences spread more readily through organizations than positive ones, and recovering from damaged first impressions requires substantially more effort than establishing initial positive perceptions through appropriate use cases and training.

Skill Gaps

Large skill gaps and learning barriers exist across the aerospace workforce, limiting widespread adoption. Even when tools are accessible, many employees face high barriers to entry—they don't know where to start or what data they can use where, so they don't begin. The "ah-ha moment" when users grasp how to apply GenAI to their work varies substantially by individual and job responsibilities, making training programs difficult to target effectively. Blue Origin provides a notable exception, expecting employees to use these tools and achieving 70% company-wide adoption [23].

Workforce Development Needs

Minimal workforce development programs exist across most aerospace organizations, with very little structured training available. Lawrence Livermore National Laboratory's aiEDGE for Innovation Day on March 26, 2025, demonstrates a possible approach—the event engaged 3,200 employees across broad job roles through demonstrations and breakout sessions focused on real-world applications, empowering staff to evaluate models for enhancing productivity and accelerating research [24]. However, training programs face substantial challenges even where organizational commitment exists. The field evolves so rapidly that information becomes outdated quickly, requiring sustained investment rather than one-time training events. Without this investment, workforce capability gaps will persist regardless of tool availability or technical maturity.

© 2026 California Institute of Technology. Government sponsorship acknowledged.

3.2.c Institutional Culture

Bureaucratic and Organizational Constraints

Overall, both GenAI suppliers and aerospace institutions (including NASA) have effectively limited bureaucratic barriers and enabled relatively rapid rollout. Organizations are providing institutional guidance on available tools and appropriate use cases in well documented websites. NASA's AI/ML site (https://nasa.sharepoint.com/sites/AIML/) and JPL's GenAI resources (https://genai.jpl.nasa.gov/) exemplify this guidance infrastructure. While procurement and security approval processes introduce delays, they have not created insurmountable obstacles to deployment.

Management Support and Expectations

Management support exerts strong influence on adoption patterns and represents perhaps the most critical cultural factor to adoption. Generally, management across organizations are supportive of employees using these tools. With that being said, there appears to be limited expectation from most organizations to push for adoption. The framing itself matters significantly—asking "Why aren't you using these tools?" creates different adoption pressure than asking "Are you using these tools?" Top-down support proves critical for any new technology adoption, and strategic pressure through management expectation can shift organizational perspective.

Project-Level Risk Tolerance

Substantial mistrust of GenAI tools persists, typical for all emerging technologies. At its core is the concern of hallucinatory behavior. The aerospace industry and, NASA in particular, have built up rigorous Verification and Validation (V&V) techniques for decades to overcome shortcomings in human reasoning and analysis; however, the trustworthiness of these new GenAI tools will require even more robust, or minimally new, V&V methodologies.

Projects and customers also exhibit general resistance to change and risk, having established processes, approaches, and expected outcomes. This conservatism constrains experimentation—teams hesitate to deploy GenAI capabilities on flight projects where stakes are highest and benefits might be greatest. The full value of GenAI augmentation may only be realized at the flight project level, yet conservative approaches limit exploration of capabilities precisely where impact would be most significant.

3.2.d Strategy

Limited Strategic Deployment Planning

Most aerospace institutions are currently focusing on rolling out infrastructure access without much attention to the broader application space. Infrastructure is an essential pillar for success, but it is getting to a point where top down strategy is required to fully realize the potential for GenAI. Some road mapping is occurring, but primarily at a grassroots level rather than from institutional leadership.

In isolated areas, NASA is beginning to develop strategic initiatives that signal growing institutional awareness of GenAI's potential. The NASA Spark platform (spark.nasa.gov) provides an internal, agency-wide collaboration space that promotes interaction across the NASA community through discussion and problem-solving, including an active Artificial Intelligence/Machine Learning community of interest. In the summer of 2025, NASA launched the "Igniting AI Innovation Across NASA" campaign to identify AI opportunities and challenges across the agency, culminating in the Ignite AI Challenge in the winter of 2026—a program designed to help the workforce learn, experiment, and build early prototypes addressing real NASA needs. NASA also established a new Small Business Innovation Research (SBIR) topic area

© 2026 California Institute of Technology. Government sponsorship acknowledged.

"AI-Powered Design for Multidisciplinary Space Hardware", explicitly targeting innovative solutions that extend AI-powered design automation of the NASA application space. The SBIR solicitation requires that proposed tools deliver design automation frameworks that accelerate hardware design, analysis, and verification while keeping human engineers in the loop for expert validation, and specifically calls for AI-based design automation tools that ingest requirements (natural language, MBSE artifacts, or standard digital files such as STEP), create designs, run embedded analyses, and return verification reports—all via open, documented APIs for compatibility with broader Text-to-Spaceship workflows. These initiatives represent NASA's recognition that strategic coordination is necessary, albeit somewhat disparate across the organization.

Finding 7

While NASA has initiated a few top-down strategic programs for GenAI in spacecraft design, greater coordinated institutional effort and investment will be required to achieve advanced capabilities such as automated spacecraft system design.

Beyond NASA, the Department of Energy is coordinating a multi-organizational GenAI initiative under the "Genesis Mission"[25]. Genesis unites DOE National Laboratories, industry, academia, and federal agency partners to harness AI for breakthroughs in energy dominance, discovery science, and national security. The White House Office of Science and Technology Policy manages cross-agency collaborations on Genesis, in which NASA Headquarters has been actively involved. Although at an early stage, this opportunity provides NASA a valuable means to contribute to the broader government GenAI mission through challenge problem definitions, along with opportunities for data and platform access and potential funding mechanisms.

3.3 Software Ecosystem Landscape

3.3.a Foundation Models Landscape

The foundation model landscape continues its rapid evolution, with substantial improvements in reasoning depth, context length capabilities, and multimodal data handling. This section provides a referential snapshot rather than comprehensive comparison, acknowledging the continuous advancement occurring across the commercial and open-weight model ecosystems. The key commercial players remain OpenAI with GPT-5.5, Anthropic with Claude 4.7, and Google with Gemini 3.1, though many other providers rapidly grow in sophistication.

Each major provider demonstrates particular strengths that guide deployment decisions. Claude shows exceptional performance for coding tasks and agentic workflows requiring extended reasoning and large context windows exceeding 200,000 tokens. ChatGPT maintains strong general-purpose reasoning and logical capabilities across diverse problem domains. Gemini leads in multimodal data processing, particularly for image and video understanding. These specializations mean that selecting among foundation models requires matching capabilities to specific application requirements rather than identifying a single "best" model.

Pricing structures vary significantly both across providers and within a single company's model offerings, with costs spanning orders of magnitude depending on model sophistication, input/output token volumes, and usage patterns (**Table 1**). For autonomous agents or multiagent systems generating substantial token volumes through iterative reasoning and tool use, computational costs can reach appreciable levels re-

© 2026 California Institute of Technology. Government sponsorship acknowledged.

quiring careful resource management. To help manage these costs and prevent unexpected cost runaways, most providers offer the ability to set spending limits on accounts over fixed time periods. Most major commercial models support deployment on government cloud infrastructure like Azure Government Cloud and AWS GovCloud, enabling their use with sensitive aerospace data subject to ITAR, EAR, and CUI restrictions, though government cloud deployments often lag commercial offerings in capability due to extended security validation requirements.

Table 1: Foundation model capabilities and pricing vary substantially across providers and model tiers, with context windows ranging from 200K to 1M tokens and costs spanning orders of magnitude. Data from pi.dev/models (accessed May 2026).

Provider	Model	Context Window	Input Cost ($/M tokens)	Output Cost ($/M tokens)
Anthropic	Claude Haiku 3.5	200,000	$0.8	$4
Anthropic	Claude Opus 4.7	1,000,000	$5	$25
OpenAI	GPT-5.4 Mini	400,000	$0.75	$4.50
OpenAI	GPT-5.5 Pro	1,050,000	$30	$180
Google	Gemini 3.1 Flash Lite	1,048,576	$0.25	$1.50
Google	Gemini 3.1 Pro Preview	1,048,576M	$2	$12

3.3.b Development Infrastructure and Frameworks

A burgeoning landscape of development tools builds upon foundation model capabilities to enable custom GenAI-enhanced applications. These frameworks support single-agent implementations through complex multiagent orchestration systems, typically integrating additional functionality such as RAG, tool use, and workflow management. Commonly used frameworks include:

- **LangChain** - Comprehensive framework for LLM applications with extensive integrations
- **LlamaIndex** - Specialized for RAG and knowledge base integration
- **CrewAI** - Multi-agent collaboration framework
- **LangGraph** - Graph-based workflow orchestration for complex agent systems

The ecosystem evolves rapidly with new players entering frequently, yet few consensus approaches have emerged for structuring these systems.

3.3.c Domain-Specific Tools

The domain-specific tool landscape demonstrates varying approaches to AI integration, with maturity ranging from proven production systems to emerging experimental capabilities. Understanding these categories and their distinct AI integration strategies helps organizations identify which tools align with current needs versus longer-term opportunities. The landscape divides into four major categories:

- Code Assistants and Development Tools
- Traditional CAE/CAD
- Workflow Automation and Orchestration Platforms
- AI-Native Engineering Tools
  - Model-Based Systems Engineering Platforms
  - Text-to-CAD and Generative Design Tools

© 2026 California Institute of Technology. Government sponsorship acknowledged.

- Multimodal Engineering Data Integration
- AI Engineering Assistants and Copilots

Code Assistants and Development Tools

Tools that augment software development through AI-assisted code generation, completion, and refactoring represent one of the most mature GenAI applications with demonstrated transformative impact across organizations. These tools were AI-native from inception, designed to leverage foundation models for deep code understanding and generation rather than retrofitting AI into existing development environments. The category has evolved rapidly from simple code completion to agentic coding systems capable of multi-file reasoning, terminal integration, and autonomous task execution including debugging and testing workflows.

GitHub Copilot pioneered this category and achieved wide adoption for reliable code completion capabilities. The category has since evolved toward more sophisticated agentic systems: Claude Code provides multi-file context understanding, terminal command execution, and autonomous debugging capabilities, while Cursor and Windsurf represent AI-native integrated development environments that embed conversational interfaces directly into the coding workflow. These tools democratize coding beyond traditional software engineers, enabling domain experts to create analysis scripts, automate data processing workflows, and build custom tools without formal programming training.

Capabilities have matured rapidly, especially for small projects like task automation scripts and moderate-sized codebases. The category continues evolving quickly, with vendors actively exploring different strategies for addressing critical challenges like context management, code compaction for large projects, memory systems for maintaining project understanding, and balancing autonomous action against human oversight. This diversity of approaches reflects rapid innovation as the field converges toward more robust solutions, despite operational maturity already achieved for specific use cases.

Finding 8

AI coding assistants represent the most sophisticated and widely adopted GenAI application beyond chatbots, with proven functionality ready for mainstream use across engineering disciplines—not merely for software engineers but as cross-cutting tools valuable for scripting, automation, and document preparation.

Traditional CAE/CAD

Established engineering platforms for CAD, physics simulation, and other Computer-Aided Engineering (CAE) platforms represent massive investments by aerospace organizations, with institutional knowledge, design libraries, and workflows built around these tools over decades. With a few exceptions, most incumbent vendors seem to be pursuing incremental, conservative AI integration strategies, with the bulk of their AI investment focusing on traditional machine

© 2026 California Institute of Technology. Government sponsorship acknowledged.

learning applications rather than GenAI capabilities. As such, the state of current CAE tools have limited to no GenAI features built in, although many are now starting to announce a more clear vision for their roadmaps and integrations. Thus, this landscape is likely to change appreciably in the intervening years.

Beyond direct GenAI integrations, successful GenAI agents will also require that incumbent CAE platforms provide robust programmatic access capabilities, which historically they have lacked. Multi-agent design systems require these tools as computational engines for physics-based analysis, requiring API access for programmatic interaction, batch execution capabilities, and automated results extraction. Traditional CAE platforms developed primarily for interactive human use through graphical interfaces often lack the robust programmatic interfaces and automation features necessary for seamless AI agent integration.

Engineering data formats present a fundamental compatibility challenge that extends beyond workflow integration concerns. The data formats foundational to engineering work—CAD geometry models, simulation meshes, technical drawings—do not fit into existing multimodal framework capabilities designed around text, images, and video. Foundation models or domain-specific GenAI technologies must develop capability to interpret and interact with these engineering data formats directly, rather than relying solely on textual or image-based representations that lose critical geometric and topological information essential for engineering reasoning.

Workflow Automation and Orchestration Platforms

Low-code and no-code platforms democratize workflow automation and agent orchestration by enabling engineers to build complex systems through visual, block diagram interfaces without traditional programming expertise. These platforms have incorporated GenAI capabilities into existing workflow infrastructure, enhancing their core automation capabilities through LLM integration while maintaining the visual, intuitive interfaces that made them accessible in the first place. This combination of visual workflow design and GenAI augmentation enables rapid prototyping and experimentation with automated workflows, allowing domain experts to iterate on automation strategies without waiting for software development resources.

The workflow automation landscape divides into two primary categories: engineering-specific platforms built for technical workflows and general-purpose automation tools that connect arbitrary applications and services. Engineering-focused platforms include Synera for CAE task automation with built-in multiagent capabilities, XMPro for digital twin software with real-time operational data integration, and Authentise for manufacturing workflow management. General-purpose platforms like n8n and Zapier provide extensive GenAI workflow support for connecting diverse applications through visual workflow builders, while Solace AI offers event-driven architecture orchestrating real-time data flow to and from autonomous agents. These platforms provide essential infrastructure for complex automated workflows, with visual interfaces making automation more approachable, intuitive, and interpretable for engineers who need to understand, validate, and maintain these systems.

© 2026 California Institute of Technology. Government sponsorship acknowledged.

Despite their visual accessibility and low barriers to entry, current workflow automation platforms lack GenAI capabilities to generate workflows themselves, still requiring manual construction through point-and-click graphical interfaces. Users must manually drag, drop, and configure workflow nodes rather than describing desired automation in natural language and having the system generate appropriate workflows. The ability for GenAI agents to create workflow definitions that automatically render as visual frameworks remains an unsolved challenge—requiring integration of conversational interfaces with code generation capabilities that produce persistent visual representations non-programmers can subsequently understand and modify.

AI-Native Engineering Tools

A new generation of engineering tools built from inception around GenAI capabilities represents a fundamental departure from incumbent platforms retrofitting AI into existing architectures. These AI-native tools design their entire workflow, data models, and user interfaces around conversational interactions and LLM-powered automation rather than adapting traditional software paradigms to accommodate AI features. The category spans systems engineering management, generative design, multimodal data processing, and AI engineering assistants—each pursuing different aspects of the engineering workflow but sharing the common characteristic of treating GenAI as foundational infrastructure rather than supplementary capability.

Model-Based Systems Engineering Platforms

Model-Based Systems Engineering (MBSE) tools manage requirements, system architecture, and interface definitions—activities central to complex spacecraft development but historically constrained by steep learning curves, specialized expertise requirements, and substantial development/maintenance overhead. The breakthrough AI integration approach adds conversational natural language interfaces to overcome the traditional MBSE complexity barriers that have plagued adoption for decades despite clear theoretical benefits. Natural language interaction eliminates the need to master complex modeling languages and tool-specific workflows, while GenAI automation dramatically reduces the time required to maintain and update these systems, potentially enabling broad MBSE adoption beyond specialty practitioners.

Most tools in this category are being built from scratch with GenAI as the foundational technology rather than retrofitting AI into legacy MBSE platforms. For example, DaVinci by Celedon provides a conversational AI agent for MBSE workflows within a unified workspace connecting requirements, system designs, simulations, and documentation. Other emerging platforms include MBSEmaestro.AI for AI-enhanced systems engineering workflows and ThunderGraph for graph-based MBSE automation. These tools are also proving useful for facilitating much of the design process, not merely requirements and architecture management.

© 2026 California Institute of Technology. Government sponsorship acknowledged.

Finding 9

AI-based MBSE platforms exhibit the clearest near-term benefit and maturity among engineering-specific GenAI tools, overcoming the traditional overhead and complexity barriers that limited MBSE adoption while proving particularly valuable for early project phases (Phase A and Phase B) and enabling dynamic capabilities beyond classical systems engineering tasks.

Text-to-CAD and Generative Design Tools

Text-to-CAD platforms enable engineers to generate parametric 3D models directly from natural language descriptions, automating the translation from design intent to executable CAD geometry. These tools target the early-stage design phase where engineers traditionally spend substantial time manually constructing geometry in CAD software, offering potential to accelerate concept exploration and enable rapid iteration on design alternatives.

The text-to-CAD category remains relatively immature with few commercial offerings available, though several companies are pursuing this capability. Zoo represents the most notable commercial platform to date, offering an AI-native CAD environment integrating GenAI into the hardware design workflow from inception. The platform automates repetitive modeling tasks, creates 3D models from text prompts, and enables geometry editing using natural language commands rather than traditional CAD operations, aiming to lower barriers to CAD proficiency while accelerating the design process.

Current implementations primarily generate scripts that execute traditional CAD operations—sketches, extrusions, fillets, patterns—rather than directly creating geometry from text, inheriting the fragility and iteration cycles characteristic of scripted CAD automation. The category faces additional challenges including limited 3D spatial reasoning capabilities and difficulty capturing complex geometric relationships in natural language. These limitations currently constrain text-to-CAD tools to relatively simple geometries and preliminary design exploration rather than detailed engineering models ready for manufacturing.

Multimodal Engineering Data Integration

Engineering work fundamentally relies on geometric and analysis data—CAD models, technical drawings, simulation meshes, and physics simulation results—that current multimodal LLM capabilities cannot process, creating demand for specialized AI platforms to address this gap. The primary bottleneck preventing native engineering understanding is lack of training data—a limitation consistently identified across academic research in this domain. Unlike text and images where massive public datasets enable foundation model training, engineering-specific data formats (parametric CAD models, simulation meshes, simulation results) remain scarce, proprietary, and difficult to collect at scale. Autodesk Research and others are exploring synthetic data generation approaches to address this scarcity, using LLMs and procedural methods to generate diverse CAD geometries and engineering scenarios for model training [26]. Additionally, a few companies are emerging to address this capability gap, focusing on developing the specialized training datasets and platforms necessary to make multimodal AI viable for engineering applications.

© 2026 California Institute of Technology. Government sponsorship acknowledged.

- Mecado (The Mechanical Data Company), delivers AI-native training data, simulations, and automation tools designed specifically for mechanical engineering, bridging the gap between AI productivity and CAD-native, physics-informed data needed to train capable engineering models.
- Pensa Technologies develops multimodal AI platforms purpose-built for manufacturing, processing diverse engineering data formats to automate analysis and decision-making tasks that traditionally required human interpretation of technical drawings, CAD models, and manufacturing specifications.

AI Engineering Assistants and Copilots

AI engineering assistants focus on work offloading and knowledge reference, targeting routine engineering tasks and rapid access to domain-specific information so engineers can concentrate on higher-level design decisions and complex problem-solving. This category represents an emerging space with limited commercial offerings as companies begin developing these capabilities. An example of software in this category is P-1 AI's Archie, an AI engineering agent targeting entry-level engineering tasks using domain-specific synthetic training data to automate cognitive engineering work.

3.3.d Ecosystem Maturity and Evolution

Maturity varies widely across the GenAI ecosystem, with certain foundational patterns stabilizing while advanced capabilities remain experimental and rapidly evolving. Understanding which aspects have achieved relative stability versus those still in flux helps organizations make informed decisions about deployment timing and risk tolerance for different categories of applications.

A majority of domain-specific software tools adopt a bring-your-own-LLM architecture, providing infrastructure for users to connect their chosen foundation model provider rather than embedding specific models within the application. This pattern offers substantial advantages for organizations: data sensitivity control enables maintaining direct relationships with model providers to ensure data handling policies align with security requirements; centralized cost management provides visibility and control over LLM usage rather than fragmented spending through multiple vendors; provider independence allows upgrading to newer models or switching providers without application vendor dependencies. For vendors, this architecture eliminates the need to maintain government cloud integrations or navigate complex LLM cost structures within licensing agreements—responsibilities that would require substantial overhead and create pricing uncertainty. The primary limitation to this approach arises when application-specific fine-tuned models are required to address gaps in current foundation model offerings, necessitating vendor-managed models or organization-specific fine-tuning infrastructure.

Certain foundational capabilities have achieved relative stability across the ecosystem. Foundation model APIs demonstrate standard patterns for text completion, embeddings generation, and function calling that have converged across major providers, enabling portable application code. Core RAG architectures for document retrieval and augmentation have matured with established approaches to vector search, document chunking strategies, and prompt template construction. Code generation capabilities exhibit high reliability for common programming tasks, with well-understood patterns for prompting, context provision, and output validation. Agentic tool use through function calling and API integration has become reliable enough for production deployment in constrained domains with appropriate oversight.

© 2026 California Institute of Technology. Government sponsorship acknowledged.

Other critical capabilities remain fluid with competing approaches and no consensus emerging. Multi-agent orchestration patterns show diverse frameworks with fundamentally different philosophies about agent architecture, communication protocols, and task decomposition strategies. Prompt engineering methodologies evolve rapidly as practitioners discover new techniques for eliciting desired model behaviors, with strategies that work well for one model generation sometimes failing for the next. Integration patterns with domain-specific engineering tools require custom approaches for each application domain, with little transferable best practice across CAD systems, simulation platforms, and other engineering software. Long-term memory and context management across extended interactions or agent lifecycles remain experimental with various approaches competing without clear winners.

This maturity gradient from infrastructure to application patterns means that organizations can confidently deploy capabilities resting on stabilized foundations while maintaining awareness that advanced orchestration and specialized integrations remain areas of active research and rapid evolution.

3.3.e Critical Gaps and Limitations

Despite rapid progress across the GenAI ecosystem, fundamental gaps spanning technical capabilities, infrastructure availability, and process maturity constrain which aerospace applications achieve deployment readiness today versus requiring breakthroughs before practical implementation.

> **Technical Gaps**
>
> Current foundation models struggle profoundly with the 3D spatial reasoning essential for physical system design—evaluating complex geometry, understanding spatial relationships between components, detecting mechanical interference, and reasoning about assembly sequences remain well beyond reliable capability. This limitation stems from two fundamental factors: training data composition and task requirements. Foundation models are trained primarily on text and images rather than engineering-specific formats like CAD geometry models, technical drawings, finite element analysis meshes, and engineering diagrams—formats that encode precise three-dimensional definitions, manufacturing intent through specialized conventions, discretized physics domains, and system relationships through domain-specific symbology. Beyond data scarcity, engineering tasks demand adherence to strict rules and requirements that differ fundamentally from natural language and image generation—geometric constraints must be satisfied exactly, physical laws cannot be violated, manufacturing tolerances must be maintained, and design decisions must trace to specific requirements rather than probabilistic pattern matching. Consequently, these native engineering formats either don't exist in model vocabularies or receive poor interpretation, forcing reliance on textual descriptions or simplified image representations that lose critical geometric, topological, and semantic information—directly impacting applications from preliminary design space exploration to automated design review. However, this represents an active area of research and commercial investment, with substantial progress expected as more diverse engineering datasets are incorporated into training, suggesting these limitations will diminish over time as the field matures.

© 2026 California Institute of Technology. Government sponsorship acknowledged.

Finding 10

Limited CAD native integration and spatial reasoning capabilities represent the most significant shortcoming in current LLM applications for mechanical engineering and other disciplines directly tied to 3D geometries, fundamentally constraining GenAI effectiveness for core engineering tasks.

Auditable traceability and reasoning transparency present significant challenges for engineering validation workflows that require understanding of how designs or analyses were developed. Current models provide limited ability to cite specific sources that influenced generation or expose reasoning chains for human review and validation—capabilities essential for engineers to verify correctness, identify errors, and build trust in AI-assisted outputs. Physics-informed reasoning reflecting conservation laws, engineering principles, and feasibility constraints remains equally limited, with models prone to generating designs that violate basic physical requirements or propose solutions incompatible with material properties and manufacturing constraints. While "thinking" or "reasoning" models exposing intermediate reasoning steps might appear to address these transparency challenges, research indicates these articulated reasoning chains are generated post-hoc to improve final output quality rather than accurately revealing the model's actual decision-making process [18] —making them unreliable for engineering validation that requires understanding true causal pathways.

Infrastructure Gaps

Most legacy engineering software lacks the APIs required for multiagent system integration, having been developed for interactive human use rather than programmatic automation. CAD platforms, finite element analysis tools, and specialized simulation software typically provide limited or no capability for batch execution, automated parameter studies, or systematic results extraction through programming interfaces. This API gap constrains the sophistication of autonomous agent workflows that could otherwise leverage these tools as computational engines.

Engineering workflows lack the data standards and infrastructure integration necessary for effective AI automation across heterogeneous tool chains. Decades of engineering tool evolution produced incompatible data formats, proprietary file structures, and absence of standardized interfaces, with no common representation enabling seamless automation across CAD, CAE, PLM, and requirements management systems. Software-focused version control and configuration management systems also do not translate to engineering artifacts, leaving tracking of CAD model changes, management of design alternatives, and maintenance of traceability through AI-assisted iterations without established solutions.

© 2026 California Institute of Technology. Government sponsorship acknowledged.

Process and Workflow Gaps

No established methods exist for systematically validating AI-generated engineering outputs, leaving organizations to develop ad-hoc review processes without guidance on appropriate rigor levels for different application criticality. Quality assurance frameworks remain uncertain—what testing is appropriate for AI-assisted designs, which review steps can safely be automated versus requiring human judgment, and how approval processes should adapt to AI integration all lack consensus answers.

A less discussed but significant challenge involves information and content overload. These tools generate vast quantities of content with minimal human effort, creating scope management challenges to keep generation focused on relevant outputs and verification/validation fatigue when humans face unrealistic volumes of AI-generated material requiring review. Without effective strategies for managing generation scope and prioritizing validation effort, organizations risk either missing critical errors or spending excessive time reviewing low-risk outputs.

Incentive Gaps and Market Maturity

The GenAI industry remains in early stages, with investment incentives naturally aligned with broad consumer markets rather than specialized domains like aerospace engineering. The cross-cutting nature of GenAI technology compounds this challenge—capabilities apply across countless domains, each with specialized workflows, data formats, and validation requirements that demand domain-specific development effort. This market reality creates persistent gaps in domain-specific capabilities: while foundation models improve rapidly for general tasks, specialized engineering applications—CAD integration, physics-informed reasoning, engineering data format understanding—advance more slowly due to limited commercial incentive to address niche technical requirements. Organizations requiring these specialized capabilities face the choice of developing custom solutions internally, waiting for gradual organic development as the broader ecosystem matures, or adapting general-purpose tools through extensive customization that may not fully address engineering-specific needs.

These gaps shape a tiered opportunity landscape for GenAI deployment at NASA. Near-term opportunities focus on high-maturity applications like code generation, technical documentation, and knowledge retrieval that avoid the most significant technical gaps and can deploy despite infrastructure and process limitations. Mid-term constraints affect moderate-maturity applications including MBSE workflows and preliminary design automation, where API limitations and integration gaps constrain capabilities more than fundamental technical deficiencies. Long-term blockers face emerging applications like fully autonomous design and sophisticated text-to-CAD generation, which require fundamental capability advances in 3D spatial reasoning, physics-informed generation, and multimodal engineering data understanding before achieving practical utility for aerospace development.

3.4 Application Areas

The cross-cutting nature of GenAI tools creates a nearly infinite opportunity space for potential applications, yet not all opportunities deliver equal value—some align poorly with current capabilities, others

© 2026 California Institute of Technology. Government sponsorship acknowledged.

solve low-priority problems, and many require breakthroughs that remain years from deployment. This section provides a systematic approach to navigating this landscape: decomposing tasks into fundamental operations that reveal whether GenAI capabilities match application requirements, then evaluating maturity and readiness through concrete examples spanning both existing commercial tools and NASA-specific applications (existing and hypothetical).

3.4.a Use Case Patterns

To describe and evaluate GenAI applications systematically, we decompose their functionality into primitive building blocks. These primitives represent common operations where language models demonstrate consistent capability, providing a vocabulary for analyzing workflows and identifying value propositions. GenAI models do not explicitly separate operations into these primitives during execution—all processing happens simultaneously within the model's forward pass. However, conceptually decomposing tasks into primitives helps practitioners evaluate whether an LLM brings value to specific workflow elements, estimate potential time savings, and identify operations where synthesis might exceed human capability.

Four primitive operations—a conceptual framework developed for this study rather than standard industry terminology—capture the core capabilities of GenAI systems.

- **Transform** operations change representation while preserving essential meaning—converting code between programming languages or reformatting technical content for different audiences exemplify this primitive.
- **Compress** operations reduce information volume while retaining key meaning, as seen when summarizing lengthy documents or extracting critical parameters from specifications.
- **Elaborate** operations expand from compact templates or outlines into detailed content with predictable structure, such as generating documentation from code comments or developing prose from bullet points.
- **Synthesize** operations generate novel analytical or creative content beyond simple template expansion—critiquing design decisions, identifying requirements gaps, or generating expert-level technical answers all require this highest-level primitive that combines reasoning with generation.

Agentic systems extend these primitives through tool orchestration and multistep autonomous workflows. Rather than executing a single LLM inference pass, agentic systems chain multiple operations together, invoke external tools, and iterate toward goals without continuous human guidance. Tool interaction itself relies on the building blocks: *compress* natural language intent and *transform* it to structured tool parameters for input processing, then apply building blocks to tool results for output interpretation. This orchestration capability enables workflows that adaptively search, synthesize, and refine across dozens of iterations, with humans defining goals upfront and validating results afterward rather than supervising each step.

3.4.b Use Case Examples

The following examples demonstrate how to decompose GenAI applications into building blocks to evaluate their deployment readiness. These examples ground the abstract capability discussion in concrete NASA-relevant scenarios, providing patterns that practitioners can adapt to evaluate applications in their own domains.

The use cases examined include:

- **Deep Research**: Multi-step autonomous web research workflows that synthesize findings across sources, now widely deployed in commercial chat interfaces

© 2026 California Institute of Technology. Government sponsorship acknowledged.

- **Agentic Coding Assistant**: Autonomous AI development assistants that execute multi-step coding tasks from natural language descriptions (e.g., Claude Code, GitHub Copilot)
- **Systems Engineering Formulation CoPilot**: RAG-based assistant providing conversational access to mission data and institutional knowledge for early mission concept development (JPL proof-of-concept)
- **ROSA**: Natural language interface to Robot Operating System enabling robot control through conversational commands (JPL proof-of-concept)
- **AiBAT**: Automated generation of Build, Assembly, and Test instructions from engineering drawings (JPL proof-of-concept)
- **Automated Engineering Drawing Compliance Check**: Multimodal AI system for automated review of engineering drawings against standards (hypothetical application)
- **Text-to-Structure**: Translation of natural language requirements directly into optimized structural CAD models and validation analyses (GSFC proof-of-concept)
- **Text-to-CAD Generation**: Natural language to parametric CAD model generation for mechanical components and assemblies

See the following sections for detailed workflow decompositions and capability analysis of each application.

> **Deep Research**
>
> Multistep research workflows autonomously search the web, synthesize findings across sources, and deliver comprehensive answers to complex questions. Similar to RAG systems but using real-time web search rather than pre-built knowledge bases, this capability represents one of the most mature and widely deployed GenAI applications. Variations of this functionality now appear in commercial chat interfaces including Google Gemini, OpenAI ChatGPT with search, and Perplexity AI, making it accessible for immediate experimentation.
>
> **Workflow Decomposition**
>
> 1. **Human:** Input research question via chat prompt
> 2. **LLM:** Synthesize research plan from user query (identify subtopics, information gaps)
>   - Optional: Human reviews/refines research plan before execution
> 3. **Autonomous iteration (3-10 cycles):**
>   - **LLM:** Compress research plan → search queries
>   - **LLM:** Transform queries → search tool call format
>   - **Tool:** Execute search → raw results (10-50 web pages per query)
>   - **LLM:** Compress results → extract key findings from each source
>   - **LLM:** Synthesize findings → evaluate gaps, generate follow-up queries
> 4. **Human re-engagement:**
>   - **LLM:** Synthesize insights across all sources (identify patterns, resolve contradictions)
>   - **LLM:** Transform findings → formatted response based on user need
>   - **Human:** Validate findings, verify critical claims, approve conclusions
>
> **Value Proposition**
>
> - **Time savings**: Research tasks requiring hours to days compress to minutes of automated search and synthesis
> - **Comprehensiveness**: Broader literature coverage than typical manual research processes

© 2026 California Institute of Technology. Government sponsorship acknowledged.

- **Accessibility**: Non-experts can conduct sophisticated research without specialized search skills or domain expertise
- **Recency**: Current information unconstrained by model training cutoff dates, critical for fast-moving technical fields
- **Example applications**: Competitive analysis, state-of-the-art surveys, precedent finding for novel designs, standards research across regulatory frameworks

Why It Works Well

- **Complex query parsing**: Excel at decomposing multi-part questions into searchable components that retrieve relevant information
- **Semantic understanding**: Interpret search results semantically rather than through keyword matching, capturing meaning even when terminology varies across sources
- **Heterogeneous synthesis**: Combine information from diverse sources (academic papers, news articles, technical documentation) without requiring uniform formatting or data structures
- **Iterative refinement**: Follow up on gaps discovered during initial searches, adapting strategy based on intermediate findings
- **Audience adaptation**: Automatically adjust presentation style for different audiences (technical expert vs. decision maker) without explicit formatting instructions

Limitations

- **Paywalled sources**: Cannot access subscription databases (IEEE Xplore, Web of Science) requiring authentication
- **Deep technical accuracy**: May miss nuances in highly specialized domains where context proves critical
- **Synthesis errors**: Can incorrectly combine contradictory information without recognizing conflicts between sources
- **Relevance filtering**: Sometimes includes tangentially related but ultimately irrelevant sources that dilute answer quality

Maturity and Deployment Readiness

This application is ready for deployment in exploratory research contexts where human validation provides quality assurance. Multiple production tools operate at scale (Google Deep Research, Perplexity, Claude with search), demonstrating high technical maturity. Off-the-shelf systems require minimal configuration beyond API access, keeping investment low. Organizations can deploy immediately for competitive analysis, literature surveys, and standards research.

Agentic Coding Assistant

Autonomous AI development assistants execute multistep coding tasks from natural language descriptions, representing a significant evolution beyond inline autocomplete tools. Unlike simple code completion systems, agentic assistants such as Claude Code and Codex can independently read codebases, plan implementations, edit multiple files, run tests, debug failures, and iterate until

© 2026 California Institute of Technology. Government sponsorship acknowledged.

tasks complete. Engineers specify goals conversationally and validate results rather than writing code line-by-line, fundamentally changing the software development workflow.

Workflow Decomposition

1. **Human:** Describe desired functionality, bug fix, or refactoring in natural language
   - Human defines scope and requirements
2. **Autonomous implementation loop:**
   - **LLM:** Compress task requirements → identify affected files and functions
   - **Tool:** Read relevant source files, grep for related code, run tests
   - **LLM:** Synthesize implementation plan → identify files to modify
   - **LLM:** Transform plan → code changes as edit operations (find/replace patterns)
   - **LLM:** Elaborate implementation with error handling, documentation, type hints
   - **Tool:** Apply edits, run tests, check compilation
   - **LLM:** Compress test failures and error messages → diagnose issues
   - **LLM:** Synthesize adjusted implementation strategy
3. **Human re-engagement:**
   - **LLM:** Present complete diff showing all changes across files
   - **Human:** Review code changes, verify correctness, approve or request modifications

Value Proposition

- **Time savings**: Significant reduction in time for routine coding tasks including data processing, testing, and documentation
- **Learning acceleration**: Junior engineers access senior-level patterns and best practices through AI suggestions
- **Boilerplate elimination**: Minutes instead of hours required for test creation, API wrappers, and configuration files
- **Exploration efficiency**: Rapidly prototype alternative implementations to compare different approaches
- **Documentation consistency**: Automatically maintain up-to-date docstrings and comments as code evolves
- **Example applications**: Rapid prototyping of analysis scripts and design automation tools, wrapping legacy C/C++ and Fortran codes for Python, generating test suites for scientific computing libraries, building custom engineering dashboards using Streamlit or Jupyter in hours versus weeks, implementing numerical algorithms from paper descriptions

Why It Works Well

- **Pattern recognition**: LLMs trained on billions of lines of code recognize common patterns, idioms, and best practices across programming languages
- **Boilerplate reduction**: Excellent at generating repetitive code structures including test scaffolding, data validation routines, and API wrappers
- **Context synthesis**: Can analyze surrounding code to maintain style consistency and architectural patterns within existing codebases
- **Cross-language translation**: Strong capability for converting implementations between languages (Python to MATLAB, C++ to Python)

© 2026 California Institute of Technology. Government sponsorship acknowledged.

- **Documentation inference**: Understands code behavior sufficiently to generate accurate explanations and usage examples
- **Error diagnosis**: Natural language understanding of compiler and interpreter error messages enables targeted fixes
- **API exploration**: Can suggest correct usage of unfamiliar libraries based on supplied library documentation

Limitations

- **Novel algorithms**: Limited ability to implement truly novel approaches not well-represented in training data
- **Complex business logic**: Multi-step workflows with intricate state management often require human decomposition
- **Security vulnerabilities**: May generate insecure code patterns (SQL injection, XSS) without understanding security implications
- **Performance optimization**: Generates functionally correct but potentially inefficient implementations
- **Numerical stability**: Can miss subtle numerical precision issues critical for scientific computing applications
- **Integration complexity**: Struggles with large-scale architectural decisions spanning many modules
- **Testing edge cases**: Generated tests may miss corner cases specific to aerospace requirements such as fault tolerance and redundancy

Maturity and Deployment Readiness

This application is ready for deployment in development of analysis tools, prototypes, and non-flight (and potentially flight) software where human validation provides quality assurance. Multiple production tools serve millions of active users (Claude Code and Codex), demonstrating high technical maturity.

Systems Engineering Formulation CoPilot

A comprehensive RAG based assistant provides systems engineers with conversational access to mission data, institutional knowledge, and formulation best practices during early mission concept development. Developed as a proof-of-concept at JPL, the CoPilot combines a curated knowledge base—including flown missions, mission studies, proposals, gate products, institutional rules, and systems engineering handbooks—with both free-form chat and guided workflows for specific formulation tasks such as analog mission search, proposal writing, heritage assessment, and science traceability matrices.

Workflow Decomposition

The system supports three primary workflows:

Free-form chat workflow (classic CoPilot style RAG workflow):

© 2026 California Institute of Technology. Government sponsorship acknowledged.

1. **Human:** Ask question about formulation, mission parameters, or institutional processes
2. **Semantic retrieval:**
   - **LLM:** Transform user query → retrieval query
   - **Tool:** Vector database semantic search → select top excerpts
   - **LLM:** Transform → augment user prompt with retrieved context
3. **Response generation:**
   - **LLM:** Synthesize grounded response incorporating multiple source excerpts
   - **LLM:** Transform → format citations with document names, relevance scores, links
   - **Human:** Review response and source citations
4. **Optional iteration:** User asks follow-up questions with preserved context

Guided workflow - Analog Mission Finder:

1. **Human:** Provide natural language mission description and search criteria
2. **Autonomous iteration loop:**
   - **LLM:** Transform natural language → structured database queries
   - **Tool:** Query mission database with filters
   - **LLM:** Compress → filter by cost/mass/timeframe ranges
   - **LLM:** Transform → generate comparison tables
   - **Tool:** Generate scatter plots and visualizations
   - **LLM:** Synthesize → rank analogs by similarity
   - User adjusts filters and re-queries
3. **Human:** Select specific analog mission for detailed investigation
4. **RAG-based mission interrogation:**
   - **LLM:** Transform mission-specific questions → retrieval queries
   - **Tool:** Vector search within selected mission documents → select top excerpts
   - **LLM:** Transform → augment user prompt with retrieved context
   - **LLM:** Synthesize detailed answers
   - **LLM:** Transform → format citations with document names, relevance scores, links
   - **Human:** Review parameters, success criteria, lessons learned

Guided workflow - Proposal Assistant:

1. **Human:** Upload Announcement of Opportunity (AO) and select proposal section to work on
2. **Requirements extraction:**
   - **LLM:** Compress → extract relevant AO requirements for selected section
   - **LLM:** Transform → structure requirements into checklist format
3. **Autonomous drafting iteration:**
   - **LLM:** Synthesize draft paragraphs responding to each requirement
   - **LLM:** Elaborate bullet points → coherent prose with technical context
   - **Human:** Review draft, provide feedback, request revisions
4. **Compliance checking:**
   - **Human:** Upload completed proposal, specify terms/concepts to check for consistency
   - **LLM:** Compress → extract all mentions of specified terms
   - **LLM:** Transform → create comparison tables showing values/claims across sections

© 2026 California Institute of Technology. Government sponsorship acknowledged.

- **LLM:** Synthesize ⟶ identify inconsistencies, contradictions, gaps
- **Human:** Review findings, resolve inconsistencies, update proposal

Value Proposition

- **Accelerated formulation**: Instant access to institutional knowledge that traditionally requires hours of manual searching across disconnected databases and document stores
- **Reduced proposal development time**: Automates requirements extraction, draft generation, and compliance checking—tasks that typically consume significant engineering hours
- **Knowledge democratization**: Less experienced engineers can leverage organizational memory and formulation best practices without extensive mentorship
- **Consistency improvement**: Horizontal review functionality catches terminology mismatches and claim inconsistencies that human reviewers might miss across long proposals

Why It Works Well

- **Proprietary knowledge access**: LLMs trained on public data cannot answer institution-specific questions about mission costs, study results, or institutional policies; RAG grounds responses in authoritative internal sources
- **Structured workflows encode best practices**: Guided workflows operationalize formulation methodologies including analog search, heritage assessment, and proposal compliance into repeatable processes
- **Multi-source synthesis**: Combines information from disconnected systems (CADRe database, data repositories, wikis, NASA handbooks) without requiring manual navigation
- **Attribution and verification**: Every response includes source citations with document names and relevance scores, enabling human fact-checking
- **Modular architecture**: Easy to add new guided workflows, update knowledge base, or connect additional data sources

Limitations

- **Limited engineering reasoning**: Can retrieve historical mission parameters but struggles to synthesize complex multi-dimensional trade-offs or recommend optimal design choices
- **Hallucination with sparse data**: When similar missions don't exist in knowledge base (e.g., first-of-kind mission concepts), may generate plausible but incorrect analogies or precedents without indication that the information is hallucinated
- **Limited multimodal understanding**: While it processes images in PDFs, it cannot deeply interpret engineering drawings, schematics, plots, or CAD models

Maturity and Deployment Readiness

This system is appropriate for exploratory research, proposal drafting, and compliance checking when all outputs receive human validation. A functional prototype incorporates multiple guided workflows, demonstrating moderate technical maturity through user testing. Investment needed is moderate—technical requirements include knowledge base curation and continuous updates, guided workflow development and refinement, custom chunking for new document types, and integration with additional data systems; testing requirements include validation against ground truth mission data, evaluation of guided workflow outputs, and security review for proprietary

© 2026 California Institute of Technology. Government sponsorship acknowledged.

data handling. This concept generalizes to a wide range of knowledge-based tasks, including virtual subject matter experts and spacecraft operations assistants.

ROSA - Natural Language Interface to Robot Operating System

An AI-powered agent provides a natural language interface to the Robot Operating System (ROS), enabling operators to interact with robots through conversational commands without requiring specialized ROS knowledge. Developed as a proof-of-concept at JPL [27], ROSA leverages LLMs and the ReAct (Reasoning and Acting) framework to translate natural language into ROS actions, executing commands through well-defined tools that wrap standard ROS utilities.

Workflow Decomposition

1. **Human:** Issue command or question in natural language (no preprocessing needed)
2. **LLM reasoning and tool selection:**
   - **LLM:** Compress user intent from natural language query
   - **LLM:** Synthesize plan of action (single tool, sequence, or follow-up questions)
   - **Human:** Verify plan
3. **Autonomous iteration loop** (if tools needed):
   - **LLM:** Transform ⟶ format tool call with parameters
   - **Tool:** Execute ROS commands, retrieve data from robot systems
   - **LLM:** Compress tool outputs ⟶ structured data
   - **LLM:** Synthesize ⟶ evaluate if query satisfied or more tools needed
   - Loop until query resolved or iteration limit reached
4. **Final synthesis:**
   - **LLM:** Synthesize ⟶ integrate observations from all tool calls
   - **LLM:** Transform ⟶ format response in natural language with appropriate technical detail
   - **LLM:** Elaborate ⟶ provide context, explanations, suggestions based on observations
   - **Human:** Review response and decide next action

Value Proposition

- **Democratized access**: Non-experts can operate ROS-based robots without command-line expertise or ROS training
- **Operational efficiency**: Faster diagnostics, monitoring, and troubleshooting through conversational queries
- **Cross-robot compatibility**: Same interface works with any ROS1 or ROS2 robot with custom tools

Why It Works Well

- **Standardized ROS structure**: ROS commands follow consistent patterns that LLMs can learn to invoke correctly
- **Tool abstraction**: Wrapping ROS utilities as Python functions with clear docstrings makes them interpretable by LLMs

© 2026 California Institute of Technology. Government sponsorship acknowledged.

- **Structured outputs**: Tools return JSON/dict data with metadata (counts, patterns, namespaces) that prevent LLM hallucination
- **ReAct framework**: Reasoning-action-observation loop enables multi-step problem solving and self-correction
- **Modular extensibility**: Easy to add custom tools for robot-specific capabilities without changing core architecture

Limitations

- **Complex multi-step queries**: Long reasoning chains can hit iteration limits or lose context across multiple tool calls
- **Ambiguity resolution**: Natural language ambiguity may require follow-up questions, increasing interaction time
- **Real-time performance**: LLM inference latency (seconds) unsuitable for time-critical control loops
- **Hallucination risk**: Despite structured outputs, LLMs may still generate plausible-sounding but incorrect responses about robot state
- **Safety validation**: Requires careful tool design with parameter validation, blacklists, and constraint enforcement to prevent unsafe commands
- **Limited physical reasoning**: Cannot reason about 3D space, mechanical constraints, or physical feasibility of commands

Maturity and Deployment Readiness

Deployment readiness depends on application context—the system is appropriate for diagnostics, monitoring, and non-critical operations with human oversight, but not for real-time control loops, safety-critical operations without validation, or environments requiring guaranteed response times. Demonstrations on multiple robots (NeBula-Spot, EELS, NVIDIA Carter) with an open-source library available show moderate technical maturity, though the system remains relatively (**Figure 5**). Investment needed is moderate—technical requirements include custom tool development for robot-specific capabilities, safety mechanism implementation, model selection and optimization, and integration with existing robot control systems; infrastructure requirements include GPU resources for local models or cloud LLM API access plus ROS/ROS2 environment setup; testing requirements include extensive validation of tool safety, parameter constraints, and failure modes; training requirements include operator familiarization with natural language interaction paradigms and understanding of agent limitations.

© 2026 California Institute of Technology. Government sponsorship acknowledged.

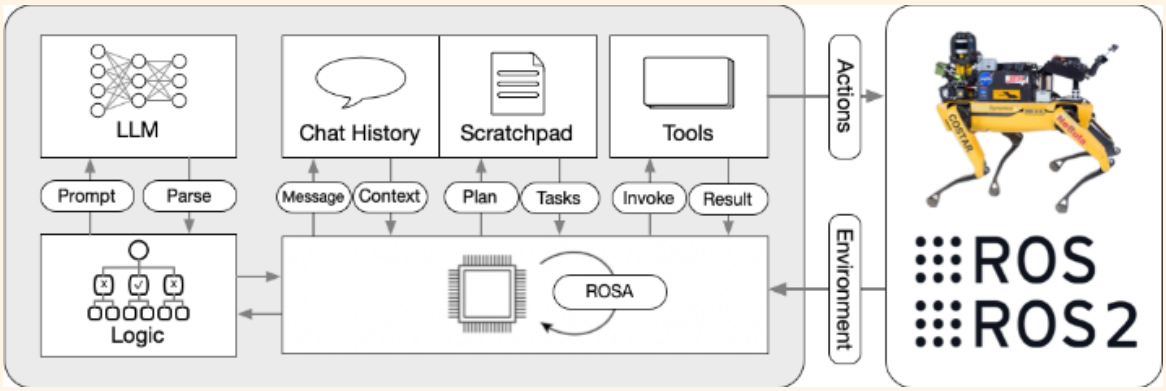


Figure 5: High-level ROSA architecture showing the integration of an LLM with ROS environments (Figure 4 from R. Royce *et al.* [27]).

AiBAT - Build, Assembly, and Test Instructions Generation

Automated generation of Instructions for Build, Assembly, and Test (IBAT) documents from engineering assembly drawings reduces manual documentation effort while maintaining quality standards. Developed as a proof-of-concept at JPL [28], AiBAT extracts information from drawing notes using Optical Character Recognition (OCR) and image processing, then uses LLMs in a multistage workflow to parse notes into structured components—actions, entities, materials—and synthesize them with standardized templates to generate technical instruction steps for hardware assembly operations.

Workflow Decomposition

> Note
>
> AiBAT was developed prior to the wide spread availability of multimodal models. Explicit tool calls to extract text and information from drawing could now likely be directly completed by a multimodal LLM.

1. **Preprocessing - Information extraction:**
   - **Tool:** Convert assembly drawing PDF ⟶ images
   - **Tool:** OCR and image processing ⟶ extract drawing notes
   - **Tool:** Detect flagged notes (triangle markers indicating importance)
   - **Human:** Verify extracted notes are complete and accurate
2. **LLM Stage 1 - Parse drawing notes:**
   - **LLM:** Compress verbose drawing notes ⟶ extract key information
   - **LLM:** Compress ⟶ identify actions (SOLDER, BOND, INSTALL, etc.)
   - **LLM:** Compress ⟶ identify entities (reference designators: C1, PRT2, U4, etc.)
   - **LLM:** Compress ⟶ identify materials and items referenced
   - **LLM:** Compress ⟶ extract process information and document references

© 2026 California Institute of Technology. Government sponsorship acknowledged.

- **Human:** Review parsed note structure for accuracy

3. **LLM Stage 2 - Generate final IBAT steps:**
   - **LLM:** Transform parsed actions ⟶ match to appropriate IBAT template sections
   - **LLM:** Elaborate ⟶ populate template with specific details (drawing number, note number, part identifiers)
   - **LLM:** Transform ⟶ format according to IBAT standards (text blocks, tables, checklists)
   - **LLM:** Compress ⟶ ensure instructions are concise and unambiguous
   - **Human:** SME review of generated IBAT steps for correctness and safety
4. **Human finalization:**
   - Quality assurance review by IBAT author's organization
   - Quality assurance review by QA organization (existing safeguard)
   - Approval and release to manufacturing/integration teams

Value Proposition

- **Time savings**: Hours per IBAT draft reduced to minutes of autonomous processing plus human review time
- **Cost**: Estimated $1.25M saved per year at JPL if deployed at scale
- **Quality improvements**: Reduces manual transcription errors from copying information between documents
- **Cognitive load reduction**: Frees engineering time from repetitive copying tasks for more challenging technical work
- **Consistency**: Automated generation ensures uniform formatting and completeness across IBATs
- **Example applications**: Printed wiring assembly IBATs, mechanical assembly IBATs, environmental test procedures

Why It Works Well

- **Clear target structure**: IBAT templates provide well-defined formatting and content requirements, reducing ambiguity in the generation task
- **Natural language understanding**: LLMs excel at interpreting technical language despite variations in how engineers write drawing notes
- **Convention learning**: Few-shot prompting effectively teaches the system organization-specific conventions without extensive retraining
- **Information transfer focus**: Task primarily involves transferring information between documents rather than generating novel engineering reasoning

Limitations

- **Template dependency**: Current system requires pre-existing "golden IBAT templates" and doesn't generalize to workflows without established templates
- **Limited scope**: Prototype only tested on Electronic Fabrication (EFAB) division use cases; mechanical and environmental test IBATs remain untested and likely more challenging
- **Missing actions and entities**: LLM sometimes omits relevant information from drawing notes, especially in unfamiliar note structures

© 2026 California Institute of Technology. Government sponsorship acknowledged.

Maturity and Deployment Readiness

This system is appropriate for assisted authoring with mandatory human review but not suitable for autonomous generation without validation. A proof-of-concept with quantitative results demonstrates moderate technical maturity, but the system is not production-ready (**Figure 6**). Investment needed ranges from moderate to high—technical improvements include better LLMs or fine-tuning, dynamic few-shot prompt generation via RAG, and fine-tuned multimodal models to handle figures, tables, and callouts found in engineering drawings; scope expansion requires testing and adaptation to mechanical assembly, environmental test, and other IBAT workflows; template generalization extends beyond golden templates to support more flexible IBAT formats; quality assurance requires establishing accuracy thresholds for production deployment and automated validation checks; integration requires connections to PLM systems, CAD repositories, and authoring tools. This concept generalizes to other fabrication areas including work order automation based on drawings for mechanical fabrication such as machining and post-processing of piece parts.

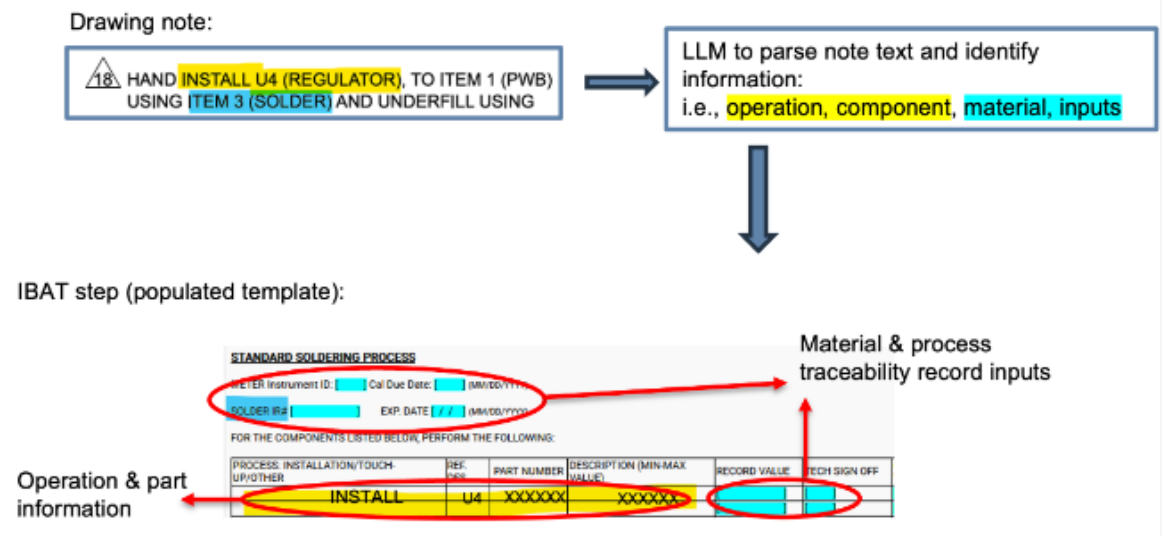


Figure 6: Conceptual workflow of AiBAT, showing how drawing notes are processed to generate IBAT documentation (Figure 2 from [28]).

Automated Engineering Drawing Compliance Check

A multimodal AI system automatically reviews engineering drawings—mechanical parts, assemblies, schematics—for compliance with institutional standards and general engineering standards including Geometric Dimensioning and Tolerancing (GD&T) per ASME Y14.5, dimensioning conventions, title block requirements, material callouts, and surface finish specifications. This hypothetical application combines computer vision to interpret drawing content with RAG-based retrieval of relevant standards, then uses LLM reasoning to identify non-compliances, ambiguities, and best practice violations that human drawing checkers would flag.

© 2026 California Institute of Technology. Government sponsorship acknowledged.

Workflow Decomposition

1. **Human:** Upload engineering drawing (PDF, TIFF, or native CAD export) and define review scope
2. **Drawing parsing and information extraction:**
   - **LLM:** Compress multimodal input → extract drawing elements (title block, geometric features, dimensions/tolerances, GD&T callouts, notes, material/finish callouts)
   - **LLM:** Transform visual elements → structured data (feature list, dimension table, GD&T specification list, notes inventory)
3. **Standards retrieval:**
   - **LLM:** Transform detected elements → formulate queries (e.g., "GD&T datum reference frame requirements", "title block mandatory fields")
   - **Tool:** RAG system retrieves relevant sections from standards knowledge base
   - **LLM:** Compress → rank and select top 10-20 standard excerpts
   - **LLM:** Transform → augment review prompt with standard requirements and examples
   - System identifies relevant standards autonomously
4. **Compliance checking loop:**
   - For each drawing element category (title block, dimensions, GD&T, notes, materials, finishes, and general layout):
     - **LLM:** Synthesize → compare extracted element against standard requirements
     - **LLM:** Synthesize → identify violations, ambiguities, missing information, best practice deviations
     - **LLM:** Elaborate → explain why issue violates standard (cite specific section/paragraph)
5. **Report generation:**
   - **LLM:** Transform → organize findings by severity (critical violations, warnings, recommendations)
   - **LLM:** Elaborate → generate detailed report with issue description/location, standard citation, non-compliance explanation, suggested correction
   - **LLM:** Transform → format as annotated PDF (issues highlighted) plus text report
   - **Human:** Drawing checker reviews automated findings
6. **Human review and disposition:**
   - **Human:** Review each flagged issue, make decisions (accept → mark for correction, reject → false positive, defer → engineering judgment)
   - **Human:** May add additional issues not caught by automation
   - **Human:** Approve drawing or return to designer with findings

Value Proposition

- **Time savings**: Conventional drawing checker review not only requires an individual's time to complete the review, but also lengthens the drawing release process
- **Reduced cycle time**: Faster initial feedback to designers means fewer review iterations and shorter drawing release cycles
- **Scalability**: Can pre-screen large drawing packages (100+ drawings) overnight, prioritizing human checker time on complex issues requiring engineering judgment

© 2026 California Institute of Technology. Government sponsorship acknowledged.

- **Quality improvement**: Catches subtle violations (datum precedence errors, incomplete tolerance chains) that human reviewers might miss
- **Documentation**: Automated reports provide clear audit trail of what was checked and why issues were flagged

Why It Works Well

- **Drawing checklists and well-documented standards**: Institutional checklists and engineering standards (ASME Y14.5, NASA-STD-8739.5, institutional drawing standards) are extensively documented with clear rules, making them ideal RAG knowledge base content
- **Pattern recognition**: Common violations including missing datums, incomplete GD&T, and ambiguous dimensions follow recognizable patterns
- **Consistent application**: AI doesn't suffer from fatigue, rushed reviews, or inconsistent interpretation—applies standards uniformly across all drawings
- **Comprehensive coverage**: Can check hundreds of requirements exhaustively (title block completeness, note formatting, dimension placement) that human reviewers might miss under time pressure
- **Natural language explanations**: LLM-generated findings include clear explanations and citations, improving designer understanding versus cryptic redline comments
- **Institutional knowledge capture**: Embeds tribal knowledge from senior drawing checkers into standards knowledge base, making expertise accessible

Limitations

- **Drawing interpretation**: Current multimodal models have limited training on engineering drawings, resulting in poor interpretation and data extraction; model fine-tuning is likely required for this application
- **Complex GD&T interpretation**: Sophisticated tolerance stack-ups, datum reference frame optimization, and compound tolerance application often require expert judgment beyond current AI capability
- **Novel or unconventional designs**: AI trained on heritage drawings may incorrectly flag legitimate novel approaches as violations when designer is intentionally deviating from convention with justification
- **Geometric reasoning limitations**: May miss violations requiring 3D spatial reasoning (e.g., impossible tolerances, inaccessible features, assembly interference) that humans visualize

Maturity and Deployment Readiness

This system is appropriate for pre-screening and assisted review with mandatory human checker validation but not suitable for autonomous drawing approval without human oversight. Multimodal vision for technical drawings is a recent capability (2023-2024) with no mature commercial tools yet, though research prototypes demonstrate feasibility, indicating low to moderate technical maturity. Investment needed is moderate to high—technical requirements include knowledge base curation from institutional and public standards (hundreds of documents), prompt engineering and workflow for drawing analysis, and false positive reduction tuning. This concept generalizes to other forms of conformance checking including contracts compliance checking, proposal compliance checking, and engineering requirements consistency and compliance checking.

© 2026 California Institute of Technology. Government sponsorship acknowledged.

Text-to-Structure

An automated system translates natural language requirements directly into optimized structural designs (CAD models) and validation analyses, eliminating the traditional iterative cycle of manual CAD modeling, finite element analysis, and design refinement. Developed as a proof-of-concept at Goddard Space Flight Center (specifics of this description vary from the actual implementation) [29], this "text-to-spaceship" approach represents an integration of natural language processing, RAG, physics-based optimization, and automated analysis.

Workflow Decomposition

1. **Human:** Provide requirements (natural language or structured data: fastener locations/orientations, load cases, mass targets, geometric constraints, manufacturing method)
2. **Autonomous requirement parsing:**
   - **LLM:** Compress natural language → extract key user parameters (bolt pattern geometry, mass budget, stiffness targets)
3. **Institutional knowledge retrieval:**
   - **LLM:** Transform mission/project context → formulate queries (e.g., "factor of safety requirements for primary structures", "thermal environment specifications for Mars missions")
   - **Tool:** RAG system retrieves from knowledge base (NASA standards, mission requirements, Evolved Structures Guide, manufacturing standards, heritage mission data)
   - **LLM:** Compress retrieved documents → extract applicable factors of safety, margins of safety, environmental load cases, manufacturing constraints
   - **LLM:** Transform → augment optimization parameters with institutional requirements
4. **Requirement synthesis:**
   - **LLM:** Synthesize user parameters + institutional requirements → complete design specification
   - **LLM:** Transform complete requirements → topology optimization inputs
5. **Autonomous design and simulation:**
   - **Tool:** Invoke preparation, topology optimization, and analysis pipeline (e.g., Synera or AI-generated automated pipeline)
6. **Results delivery:**
   - **LLM:** Elaborate → generate human-readable analysis report (stress summaries, mode frequencies, manufacturing feasibility notes, mass/stiffness vs. target)
   - **Human:** Review design, analysis results, manufacturing feasibility

Value Proposition

- **Faster design cycle**: Replaces human-in-the-loop process (sketch CAD → simulation → redesign) requiring multiple engineers to iterate with a fully automated process
- **Democratized optimization**: Less experienced engineers can generate expert-level optimized structures without deep CAD expertise, topology optimization knowledge, or simulation analysis skills
- **Superior design performance**: Automated designs consistently achieve 20-60% mass reduction versus human-designed equivalents while meeting or exceeding stiffness/strength requirements

© 2026 California Institute of Technology. Government sponsorship acknowledged.

- **Higher level of design**: The potential to expand to assemblies of structures, leading to the automated design and validation of subsystems or even the full spacecraft structure
- **Offline design**: While the design process is running, engineers are freed up to work on other tasks

Why It Works Well

- **Physics-based design**: Topology optimization is physics-driven, ensuring structurally valid geometries that satisfy conservation laws and equilibrium conditions
- **Institutional knowledge accessible via RAG**: Design standards (NASA-STD-5001), environmental requirements, and best practices stored in searchable knowledge base
- **Generative design maturity**: Commercial topology optimization tools (Fusion 360, nTopology, Altair OptiStruct) are production-ready with robust solvers, well-validated simulation engines, and manufacturing constraint handling
- **Narrow, well-defined problem**: Structural brackets and frames have clear success metrics (mass, stiffness, stress margins, frequencies) unlike open-ended design problems requiring subjective judgment

Limitations

- **Limited design cases**: Current commercial topology optimization tools are generally limited to purely structural design problems; much of spacecraft hardware is also driven by thermal conditions (this is not a fundamental limitation of topology optimization, but reflects current functionality in commercial tools)
- **Design and validation robustness**: Topology optimization (design convergence) and automated finite element simulation (such as meshing) have limited robustness and are prone to fragile implementations that can be hard for current GenAI models to troubleshoot
- **Manufacturing reality gap**: Generative design "manufacturability" constraints are idealized and don't necessarily cover all process limitations
- **Assembly context missing**: Optimizes individual parts in isolation; cannot reason about assembly sequence, fastener access, integration tooling, or how structure interacts with adjacent components
- **Iteration cost**: Each design and simulation has non-trivial computation time; exploring large design spaces requires significant computational resources and time

Maturity and Deployment Readiness

The technology shows promise for structural component design but requires significant development before achieving production readiness for complex assemblies or multi-physics design scenarios. A minimum viable product demonstrated through the Boop! project at Goddard Space Flight Center indicates low to moderate technical maturity (**Figure 7**). Investment needed ranges from moderate to high—requirements include topology optimization tools with greater capability to solve design problems related to all NASA load cases and conditions (such as thermal and dynamic load cases and thin-walled structures) and less fragile finite element analysis techniques (such as mesh-free methods which aren't susceptible to mesh failures).

© 2026 California Institute of Technology. Government sponsorship acknowledged.



Figure 7: Goddard Space Flight Center's Boop! payload on a high altitude balloon, demonstrating a minimum viable product for Text-to-Structure. Image credit: NASA GSFC.

Text-to-CAD Generation

Natural language to parametric CAD model generation enables engineers to describe mechanical components, assemblies, or systems through conversational text and receive editable CAD geometry. Rather than manually constructing features in CAD software, engineers specify design intent, constraints, and requirements through dialogue with GenAI systems that generate parametric models.

Workflow Decomposition

1. **Human:** Provide text and/or images describing component geometry, function, constraints, materials, manufacturing considerations (e.g., "Create a mounting bracket for a sensor box, 100mm × 80mm footprint, M6 mounting holes, must clear 20mm tall connector on bottom face")
2. **Design intent interpretation:**
   - **LLM:** Compress natural language ⟶ parse geometric features, dimensions, constraints, relationships
   - **LLM:** Synthesize ⟶ identify ambiguities or missing information
   - **LLM:** Transform ⟶ generate internal representation of design requirements
3. **Autonomous CAD generation and validation loop:**
   - **LLM:** Transform ⟶ generate parametric CAD script (Python for CAD API, OpenSCAD code, or neutral STEP with parameters)
   - **LLM:** Elaborate ⟶ define features sequentially (sketches, extrusions, holes, fillets, patterns)
   - **LLM:** Elaborate ⟶ include dimensional parameters and relationships

© 2026 California Institute of Technology. Government sponsorship acknowledged.

- **Inner loop - Script execution and debugging:**
  - **Tool:** Execute CAD script to generate geometry
  - **LLM:** Compress → analyze script execution errors (syntax errors, geometric failures, constraint violations)
  - **LLM:** Synthesize → diagnose issues and adjust script (fix feature ordering, resolve constraint conflicts)
  - Loop until script runs without errors
- **Tool:** Export rendering/image of generated geometry
- **LLM:** Compress → multimodal validation of geometry against original design intent (visual inspection of features, proportions, orientations)
- **LLM:** Synthesize → identify discrepancies between intent and actual geometry (missing features, incorrect dimensions, wrong orientations)
- Outer loop repeats entire generation → debugging → validation process until design intent achieved

4. **Human re-engagement for model review:**
   - Engineer visualizes geometry in CAD environment
   - Engineer checks dimensional accuracy, feature relationships, design intent capture
   - **Human:** Identify issues requiring correction
5. **Human-AI refinement loop:**
   - **Human:** Provide refinement instructions (e.g., "Increase wall thickness to 3mm, add 2mm fillet to all internal corners")
   - **LLM:** Compress modification requests → parse requirements
   - **LLM:** Transform → update parametric script with changes
   - Re-execute step 3 (autonomous CAD generation and validation loop) on updated script
   - Loop repeats refinement and validation until design intent fully captured

Value Proposition

- **Accelerated concept exploration**: Rapidly generate multiple design alternatives from text descriptions without manual CAD modeling time
- **Reduced CAD expertise barrier**: Junior engineers or non-CAD specialists can create geometry for analysis or visualization without deep tool proficiency, democratizing access to 3D modeling capabilities
- **Design documentation alignment**: Natural language descriptions become living documentation synchronized with geometry
- **Parametric by default**: GenAI-generated scripts naturally include parameterization, enabling design space exploration and optimization without additional parameterization effort

Why It Works Well

- **Code generation capability**: Parametric CAD scripts follow predictable patterns and syntax that align with GenAI's proven code generation strengths
- **Iterative refinement**: Dialogue-based workflow matches GenAI's conversational interface, allowing progressive design development through natural interaction
- **Parameter extraction**: LLMs effectively identify and structure numerical values, dimensional relationships, and constraint hierarchies from unstructured text

© 2026 California Institute of Technology. Government sponsorship acknowledged.

Limitations

- **Natural language limitations for complex geometry**: Simple features like bolt patterns are straightforward to describe, but complex geometric features (fillet blends, curved surfaces, compound angles) are difficult to specify unambiguously in natural language
- **Complex topology**: Non-trivial surface modeling, organic shapes, and intricate geometric relationships exceed current GenAI geometric reasoning capabilities
- **Fragile CAD automation**: CAD operations are inherently fragile and difficult to automate reliably—feature ordering dependencies, geometric solver failures, and constraint conflicts require extensive iteration; scripts may fail to converge to valid geometry, necessitating repeated debugging cycles
- **Limited geometric reasoning**: Cannot perform spatial reasoning about interference, load paths, or functional performance without explicit instruction; cannot reliably verify manufacturability
- **Limited CAD API ecosystem**: CAD geometry kernels are proprietary with platform-specific APIs—Python scripts cannot create parametric models compatible across platforms (e.g., script for FreeCAD won't work in NX); API documentation often not publicly available or in LLM training data; CAD operations and terminology vary between platforms
- **Verification burden**: Engineers must carefully validate all dimensions, relationships, and design intent capture; errors may be subtle and hard to detect

Maturity and Deployment Readiness

The technology shows promise for simple component generation but requires significant development before achieving production readiness for complex parts or reliable integration into aerospace workflows. Proof-of-concept demonstrations exist for basic components including brackets, mounts, and housings through research prototypes at universities [30], [31] and early commercial tools (Zoo), indicating low to moderate technical maturity with substantial manual prompt engineering and iteration required for useful results. However, this represents an active area of research and commercial development, with capabilities expected to improve substantially in the coming years as multimodal models advance in geometric reasoning and CAD vendors expand API capabilities.

© 2026 California Institute of Technology. Government sponsorship acknowledged.

4 Pilot Demonstration Study

We conducted a hands-on demonstration study to achieve a more grounded understanding of GenAI capabilities applied to spacecraft design. With most technologies, the devil lies in the details—promising high-level capabilities can encounter show-stoppers or implementation hurdles that limit practical functionality. The information we sought centered less on the final product and more on the lessons learned from applying this technology toward a representative design goal.

4.1 Pilot Design

4.1.a Case Study Definition: Design Scenario

The Mars Sample Return (MSR) campaign represents a multi-mission effort to collect scientifically selected samples from the Martian surface and return them to Earth for detailed laboratory analysis [32]. NASA's Perseverance rover has been collecting and caching sample tubes since its February 2021 landing [33]. One proposed MSR architecture included a pair of Sample Recovery Helicopters (SRH) to retrieve these cached samples and transport them to a Sample Retrieval Lander for return to Earth [34]. This helicopter-based architecture built upon the successful demonstration of powered flight on Mars by the Ingenuity helicopter, which accompanied Perseverance as a technology demonstration (**Figure 8**).



(a) Ingenuity helicopter [35] (b) SRH concept [34]

Figure 8: Mars helicopter designs showing the evolution from the Ingenuity technology demonstrator to SRH. The SRH design builds upon Ingenuity's proven architecture, adding sample recovery mechanisms, enhanced avionics, larger rotors, and increased payload capacity to enable sample tube retrieval and transport during the Mars Sample Return campaign.

Our case study design scenario involved using GenAI to assist in designing SRH to Preliminary Design Review (PDR) fidelity. NASA PDR level design is sufficiently mature to establish requirements allocation, preliminary subsystem sizing and interfaces, key technical risks, and analytical models demonstrating feasibility to proceed to detailed design. We leveraged the existing internal and externally published design corpus of the successfully designed Ingenuity helicopter [35], [36], [37], [38], [39], [40], [41], [42]. This heritage design, along with standard NASA and JPL institutional best practices and requirements, and more general engineering best practices, served as the knowledge base for the study.

4.1.b Case Study Selection Rationale

We selected SRH as a case study because it represented a relevant NASA mission with tractable scope and complexity for evaluating GenAI design capabilities. JPL has worked on multiple Mars helicopter design variations, ranging from Ingenuity (fully designed, built, and operated on Mars) [35], [38], [39], to SRH

© 2026 California Institute of Technology. Government sponsorship acknowledged.

(designed to PDR level until the mission architecture pivoted away from helicopters), to the much larger Chopper design concept [43]. SRH provided an ideal case study because its base architecture derived from the Ingenuity helicopter and a full preliminary design had been completed, providing a baseline reference for comparison with conventional design approaches. Helicopters are also common in terrestrial applications, making it likely that their physics and relevant features already exist within LLM training sets. However, flying on Mars presents notably different flight conditions, making this an interesting test of extrapolating core physics to different environments.

4.1.c Tool Selection and Configuration

Creating custom tooling for this pilot proved impractical given the project scope and budget. We also recognized value in choosing a commercially available tool to investigate the state of practice, particularly since broad implementation of this technology will likely require commercial options rather than bespoke organizational tools. We selected Celedon's Davinci tool for its high-level capabilities, multiagent framework, and adaptability to a wide range of tasks. As an MBSE tool, Davinci is well suited for preliminary design, providing an ecosystem to both represent an engineering system and integrate modeling, simulation, and other tool capabilities to ground the design in real physics.

We completed all work using Claude's Sonnet 4.5 via government cloud. Sonnet was chosen both due to its accessibility within JPL (in particular, for sensitive data) and its well known effectiveness at agentic and multiagent reasoning and orchestration.

4.1.d Approach

Our study team consisted of diverse backgrounds, including structures, thermal, mechanisms, robotics, aerodynamics, and systems engineering. Two team members had worked on SRH previously, but the remainder had zero exposure to the design. Given such a new process, our approach remained primarily exploratory within the context of the objectives rather than prescriptive. Thus, much of the pilot consisted of understanding effective ways to interact with and use GenAI for this engineering task.

The primary work for this pilot study was conducted in the context of collaborative hackathons. We conducted three hackathons total (two full day and one half day) where the full team of seven people came together in a single space to facilitate this learning process. Each hackathon had targeted goals, with team members working individually on domain-specific problems and periodically regrouping to share what was working and what wasn't, both in scheduled sessions and ad hoc throughout each day.

1. The first hackathon focused on familiarizing the team with Davinci and understanding how best to approach the design problem with the tool.
2. The second hackathon, held a few weeks later, applied lessons learned from the first session along with intermediate individual work to more task-oriented, discipline-specific problem solving with Davinci.
3. The third hackathon, held a few weeks after the second, focused on refining the various processes that proved most successful in earlier iterations.

In between each hackathon session, team members continued to work on their respective tasks and design assignments.

4.2 Results and Discussion

Ultimately, we did not develop a fully PDR-level design of an SRH helicopter within the pilot study scope; however, the pilot achieved its primary objective: understanding GenAI capabilities and implementation challenges through hands-on application. This outcome reflects several factors: the inherent complexity

© 2026 California Institute of Technology. Government sponsorship acknowledged.

of even a drone scale helicopter system, limited project scope and budget, current tool limitations, and the team's learning curve with these novel design methodologies. The study revealed specific high-value capabilities that work well today, critical limitations requiring careful management, and effective practices for human-AI collaboration in engineering design.

4.2.a Demonstrated Capabilities

Preliminary Design Dashboards

AI coding tools dramatically reduced the barrier to creating interactive design exploration dashboards (**Figure 9**). Several discipline experts used Davinci and Claude Code to generate the complete application code—both backend physics calculations and frontend user interfaces—for relatively sophisticated dashboards exploring structural sizing based on beam theory and robotic arm kinematics, including reach area analysis and obstacle avoidance. Traditionally, engineers avoid building such tools because the effort required—days of work for a skilled programmer—far exceeds the value for single-use analysis. GenAI changed this calculus: dashboards that would require days of manual coding now take hours to develop with AI assistance, making them practical for routine design exploration. Because these dashboards serve individual engineers rather than requiring broad distribution, they need not be polished production tools, and the underlying physics can be readily verified by subject matter experts familiar with the domain.

© 2026 California Institute of Technology. Government sponsorship acknowledged.

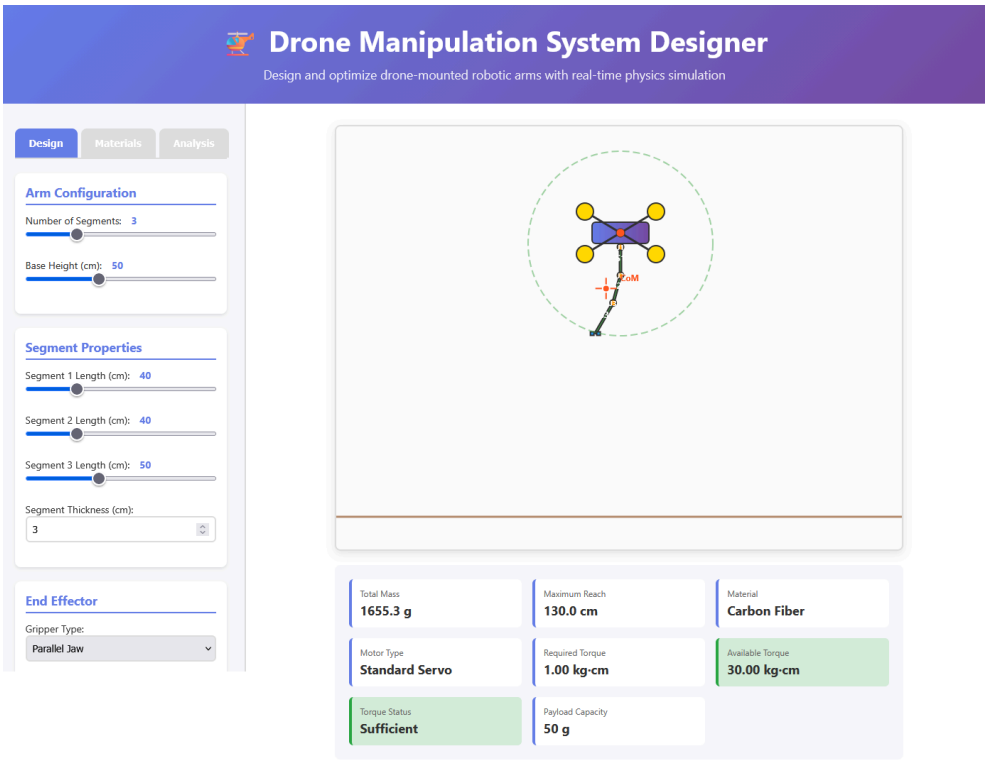


Figure 9: Interactive design dashboard for robotic arm exploration generated using GenAI coding tools. The dashboard enables real-time design space exploration of the SRH robotic arm, including arm configuration, segment properties, and end effector selection. Key metrics displayed include arm reach, mass, and required torque, allowing engineers to rapidly evaluate design trade-offs for stability and payload capacity.

Preliminary Analysis

Beyond interactive dashboards, AI coding tools demonstrated capability in building more sophisticated analysis implementations from source documentation. For example, the rotor design team provided the system with technical papers describing blade element momentum theory analysis techniques alongside experimental validation data from Ingenuity flight tests. Through iterative refinement, the AI generated simulation code calculating aerodynamic performance metrics including thrust, power requirements, and efficiency across various operating conditions (**Figure 10**). The system automatically validated the generated code against the provided test data as part of the development process, confirming that the implementation correctly captured the underlying physics and matched expected performance trends. This capability could then be used for preliminary sizing exercises, either integrated into interactive design dashboards for real-time exploration or used in more rigorous optimization workflows to converge on rotor configurations meeting mission requirements.

© 2026 California Institute of Technology. Government sponsorship acknowledged.

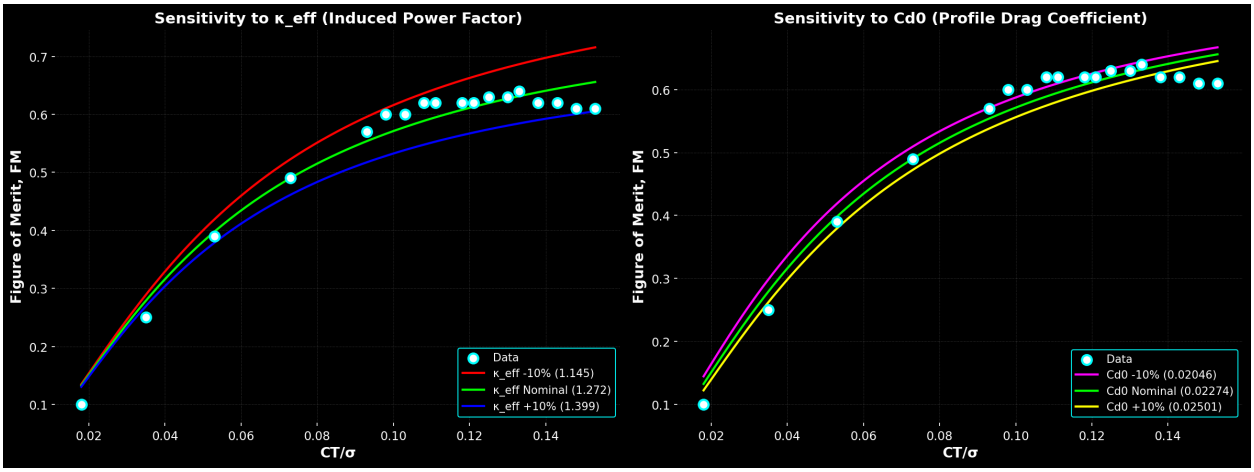


Figure 10: Rotor blade element momentum theory sensitivity analysis generated using AI coding tools. The analysis shows thrust, power, and efficiency variations across different rotor operating conditions, validated against Ingenuity flight test data. Such automated analysis generation enables rapid design space exploration during preliminary sizing.

Finding 11

GenAI coding agents are highly effective at automating preliminary engineering analysis, including calculations for sizing and performance prediction, along with the ability to rapidly create dashboards that provide a more intuitive interface to the interactions of various elements of a system or design.

Requirements Derivation from Heritage Missions

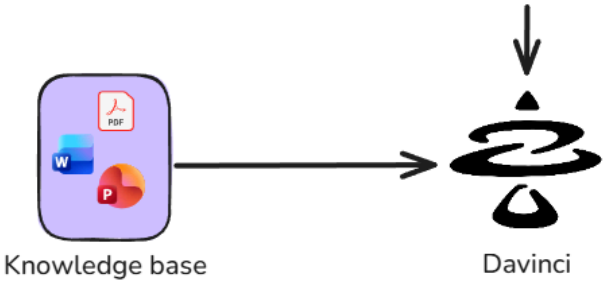
GenAI demonstrated strong capability in deriving mission requirements from heritage designs (**Figure 11**). Given the Ingenuity helicopter knowledge base and a high-level description of SRH mission objectives, the tools generated updated and new requirements that showed good consistency when compared against the previously developed SRH requirements baseline. This represents a significant efficiency gain: what would typically require a systems engineer several days of careful analysis was accomplished in hours through iterative refinement with the AI tools. The quality of AI-generated requirements suggests these tools can accelerate early mission formulation while maintaining technical rigor, provided expert guidance shapes the generation process.

Finding 12

Systems engineering tasks—requirements derivation, management, and analysis—represent readily achievable GenAI applications, as these language-based activities require contextual understanding across diverse sources, a capability where LLMs excel.

© 2026 California Institute of Technology. Government sponsorship acknowledged.

"I have uploaded design documents to your Library describing the Mars Helicopter, Ingenuity. We want to build a new mission, Sample Recovery Helicopter (SRH), that will build upon the Ingenuity design to be able to pick up sample tubes dropped on the surface of Mars by Perseverance Rover. SRH will be deployed from a new Lander (SRL), pick up tubes, and drop them beside the SRL Lander. Provide insights into key additional Ingenuity capabilities required to augment and fly SRH."



Required Augmentations

1. Sample Handling Mechanism (CRITICAL) - Ingenuity has zero manipulation capability; SRH needs a complete gripper system (~100-200g)
2. Precision Navigation (CRITICAL) - Current 2-3% positioning accuracy must improve to <0.3% for centimeter-level tube pickup
3. Payload Capacity (MAJOR) - 165-315g mass increase (9-17%) requires rotor performance analysis and potential upgrades
4. Power & Energy (MAJOR) - Need 50-60 Wh battery (vs. 42 Wh) and extended flight time to 200-300 seconds
5. Autonomy (MAJOR) - Transform from pre-programmed sequences to autonomous tube search, grasp verification, and retry logic
6. Operational Complexity (MAJOR) - Multi-phase sorties (search→pickup→delivery→return) vs. simple point-to-point flights
7. Flight Performance (MODERATE) - Extend thermal-limited 170s flight time and develop precision hover modes
8. Structural Robustness (MODERATE) - Reinforce for gripper loads and multiple landing cycles

Figure 11: Initial assessment completed to understand the "required" augmentations to the Ingenuity design to meet the SRH requirements. These augmentations were then compared with the actual SRH changes and showed good consistency (although were not identical and some were purely hallucinated).

© 2026 California Institute of Technology. Government sponsorship acknowledged.

4.2.b Limitations and Challenges

Hallucination Despite Knowledge Systems

Currently, curated knowledge systems reduce but do not eliminate hallucination. The insidious challenge is that fabricated content—a dimension, a material selection, a design parameter—presents with the same confidence as grounded information, offering no indication of its synthetic origin. Direct prompting helps: explicitly instructing the system to cite sources and acknowledge uncertainty when information is unavailable. However, even vigilant prompting cannot prevent all hallucinations, particularly when the system synthesizes across multiple sources or abstracts beyond explicit source content. This demands continuous verification: engineers must validate all generated content against source materials and domain expertise, treating AI outputs as drafts requiring expert review rather than authoritative references.

The requirements gap assessment shown in **Figure 11** illustrates this challenge concretely. The AI correctly identified several Ingenuity design constraints from source documents: no manipulation capability, flight time limited by thermal constraints, point-to-point flight operations, and the 42 Wh battery capacity. However, other values—particularly numerical values like gripper system mass, payload capacity, and specific flight duration—were incorrect or hallucinated, though remaining within plausible ranges. These are not values that can be derived from text prediction alone; they require physics-informed calculations, engineering models, or simulation tools operating on the backend to produce accurate results. The challenging aspect is distinguishing which numbers came from actual sources versus which the system fabricated to complete the analysis. Without exhaustively tracing each claim back to source documentation, engineers face uncertainty about every generated value. These systems rarely acknowledge what they don't know, instead confidently filling gaps with synthesized but ungrounded information, necessitating verification of all outputs regardless of apparent confidence.

Verification and Validation Fatigue

The verification burden imposed by AI-generated content emerged as a significant practical challenge. Traditional engineering relies on trust developed through professional relationships and demonstrated expertise—colleagues earn credibility through repeated accurate work. AI tools start from zero credibility, rightfully requiring rigorous verification of every output. The asymmetry is stark: AI generates immense volumes of content in minutes, while human verification proceeds at the slower pace of careful review. This creates a backlog of unglamorous verification work that can quickly become overwhelming, tempting engineers to defer validation or skip corrections in favor of more engaging tasks. Alternatively, the verification burden can discourage adoption entirely—engineers reason that if they must rigorously check every output anyway, they might as well generate the content themselves rather than spending time correcting AI errors.

The incremental tasking approach described below helps manage this burden by limiting output volume per interaction. Looking forward, AI coding has demonstrated potential for automated verification through agent-generated tests, but coding's well-defined correctness criteria (does it compile? do tests pass?) do not translate directly to open-ended engineering design problems. As

© 2026 California Institute of Technology. Government sponsorship acknowledged.

adoption grows, engineers will likely spend increasing time defining the validation protocols and unit tests that enable more automated verification. This remains a "guilty until proven innocent" paradigm for the foreseeable future—necessary caution that imposes real cognitive load on practitioners.

Finding 13

The combination of current tool maturity levels and high accuracy requirements for engineering applications necessitates extensive verification of GenAI outputs—while RAG systems and agent tool usage provide grounding, verification and validation remains a significant and non-trivial workload.

4.2.c Effective Practices

Incremental Tasking Over Large-Scope Prompting
As has been found with AI coding agents, the most effective workflow pattern proved to be incremental tasking—decomposing complex design problems into smaller, verifiable steps rather than requesting end-to-end solutions. This reflects an evolving understanding of prompt engineering, where human intuition about how to ask questions often diverges from what produces reliable AI behavior.

The natural human approach is to pose the complete problem: "Based on the Ingenuity helicopter, design me a new helicopter capable of retrieving Martian samples on the order of XXX grams at a range of XXX meters." This appears well-specified but conceals the problem—insufficient context exists to actually design a viable helicopter, requiring thousands of engineering decisions beyond the stated parameters. The LLM will not acknowledge this insufficiency. Instead, it confidently generates an extensive design, pulling some elements from the knowledge base while hallucinating plausible-sounding values to fill gaps. The result appears credible on first inspection but fractures under scrutiny, requiring more effort to correct than starting fresh. Even meticulously detailed prompts cannot prevent subtle derailments when operating at large scope.

Incremental tasking addresses this by maintaining human oversight at decision points. Rather than "design a Mars sample retrieval helicopter," an effective sequence might be: "What are the critical subsystems for SRH?", followed by "For the rotor assembly, what are the required components?", then "What are the mass and power requirements for each component?" AI agents can assist in developing these incremental task breakdowns themselves, helping engineers identify appropriate decomposition strategies for complex problems before execution begins. Each response receives validation before the next question builds upon it, keeping the design process grounded while still leveraging AI capabilities for execution within bounded tasks.

© 2026 California Institute of Technology. Government sponsorship acknowledged.

Revision Control for Design Management

AI-generated design changes happen fast—sometimes too fast to track without explicit version management. When an LLM executes sweeping modifications across a design model, understanding exactly what changed becomes difficult without deliberate tracking mechanisms. Software engineering solved this problem decades ago with git and other version control systems, but these practices have not penetrated deeply into other areas of engineering.

Davinci's implementation of revision control within their MBSE framework proved highly effective, providing clear visualization of additions, modifications, and deletions in the design model. More importantly, branching enabled risk-free exploration: maintaining a stable baseline design while spinning off branches to explore alternatives without contaminating the primary design. Successful concepts merge back; unsuccessful ones are abandoned without consequence. This pattern will become crucial as teams grow and as verification requirements demand clear audit trails of design evolution. Historically absent from general engineering design practice, revision control emerged from this pilot as a valuable discipline to import from software development—particularly valuable when AI can generate changes faster than humans can absorb them.

Finding 14

GenAI-accelerated content generation demands rigorous version control extending beyond software to all engineering data products, as AI produces design changes faster than humans can fully oversee.

Report-First Workflows for Design Intent Capture

Similar to planning mode in AI coding tools, generating and iterating on the design approach and method was found to be an effective way to not only guide design generation, but also to provide a means to capture the design intent and methodology. When facing an analysis or sizing exercise, the instinct is to jump directly into implementation—write the script, run the numbers, iterate until it works. The report-first approach reverses this: use GenAI to first draft an analysis plan documenting scope, assumptions, and methodology. This intermediate artifact is far easier to audit and verify than code, provides structure to guide subsequent implementation, and creates a readable explanation that other team members can understand without parsing through equations and conditionals. Critically, this documented approach then feeds directly into the incremental tasking strategy described above, providing a validated roadmap that can be decomposed into specific, bounded tasks for AI execution. As a secondary benefit, this practice helps manage context window limitations by externalizing mature design knowledge into documents that can be retrieved through RAG rather than maintaining everything in active conversation context.

This addresses a persistent gap in engineering practice: design intent remains largely undocumented. Engineering designs get frozen at gate reviews—PDR, CDR—which capture what was built and whether it meets requirements, but not why particular choices were made or what alternatives were considered and rejected. Report-first workflows naturally generate this missing context: the

© 2026 California Institute of Technology. Government sponsorship acknowledged.

> rationale, the trade-offs, the assumptions that shaped decisions. These documents serve not just the current project but future missions facing similar challenges, particularly as GenAI-based knowledge systems make historical design rationale conversationally accessible to future teams. The long-term institutional value could be substantial, transforming how engineering knowledge accumulates and transfers across missions.

© 2026 California Institute of Technology. Government sponsorship acknowledged.

5 Recommendations and Path Forward

The assessment of GenAI capabilities, organizational readiness, and application maturity across aerospace reveals a technology landscape at an inflection point. Technical capabilities have advanced sufficiently to deliver immediate value in specific domains while pointing toward transformative long-term potential. However, realizing this potential requires strategic investment choices that recognize both what GenAI can accomplish today and what fundamental capabilities remain under development. The question facing NASA is not whether to adopt GenAI, but how to lead its responsible deployment across the aerospace ecosystem—leveraging the agency's deep technical expertise to validate capabilities, establish best practices, and enable industry partners to adopt these tools with confidence. These recommendations provide actionable guidance for NASA to capitalize on near-term opportunities while positioning for future breakthroughs.

5.1 Strategic Investment Principles

NASA faces critical decisions about where to invest limited resources in a rapidly evolving technology landscape. A few foundational principles should guide these investment choices.

5.1.a Leverage Rather Than Duplicate

Substantial work developing GenAI capabilities is already occurring outside NASA across industry, academia, and the broader AI ecosystem. Industry is already investing billions of dollars in training frontier foundation models—OpenAI, Anthropic, Google, and others compete intensely to advance core LLM capabilities. Minimal return on investment exists for NASA to build competing foundation models when commercial offerings provide state-of-the-art performance with flexible deployment options including government cloud infrastructure. NASA should leverage commercial foundation models as computational engines rather than duplicating commercial development, concentrating investment instead on NASA-specific capabilities that won't emerge organically from the commercial ecosystem: aerospace knowledge bases, engineering toolchain integration, workflow adaptation, workforce training, and infrastructure access. Fine-tuning or specialized model development makes sense only where NASA's unique requirements —ITAR-controlled design data, mission-specific operational contexts, specialized aerospace terminology —create needs that commercial providers cannot address through general-purpose models.

Finding 15

Significant GenAI development investment already exists across broader industry in areas where NASA duplication offers minimal return—NASA must strategically identify ecosystem gaps critical for NASA applications that will not emerge naturally from current commercial investments.

5.1.b Match Strategy to Application Scale

Different scales (size and scope) of GenAI applications require fundamentally different organizational approaches to achieve successful adoption.

- *Small-scale* applications including coding tools, documentation assistants, and individual productivity enhancements will likely happen organically through grassroots adoption—engineers need these tools to work more efficiently and will adopt them when organizational barriers (access, security approval, basic training) are removed. These applications require modest organizational support: provide approved tool access, clear guidance on appropriate use cases and data sensitivity constraints, and

© 2026 California Institute of Technology. Government sponsorship acknowledged.

lightweight training to accelerate learning curves. Individual champions and peer-to-peer knowledge transfer will drive adoption once fundamental access barriers are eliminated.

- *Large-scale* applications present fundamentally different challenges. Subsystem-level design automation, integrated multi-disciplinary workflows, and AI-augmented development processes require development of new organizational practices, strategic investment in supporting infrastructure, and careful change management. These won't happen organically—people naturally prefer existing methods they understand, and will identify legitimate limitations in new approaches that create hesitation about adoption. Strategic implementation of large-scale applications requires both substantial investment and active top-down support, with leadership clearly signaling where and why GenAI makes sense rather than mandating AI integration into every process. Leadership must provide explicit encouragement that these technologies are not only acceptable but expected for appropriate use cases, creating organizational permission and expectation for adoption. Without visible top-down support and resource commitment, large-scale applications will stall at the pilot stage regardless of technical maturity.

NASA should actively pursue both small-scale and large-scale GenAI adoption, recognizing they represent different strategic opportunities. Small-scale applications are already gaining traction organically across the agency—*success here depends on maintaining robust infrastructure, staying ahead of workforce needs, and systematically enabling engineers to leverage these productivity tools.* Large-scale applications present transformative potential for how NASA conducts spacecraft development, but demand a more strategic approach: *targeted guidance identifying where GenAI delivers genuine value, direct investment in enabling capabilities, and sustained top-down support that signals organizational commitment.* Both are critical to NASA's future competitiveness—small-scale adoption builds workforce capability and organizational familiarity, while large-scale applications position the agency to fundamentally improve cost, schedule, and technical performance in spacecraft development.

5.2 Specific Recommendations

The following recommendations provide concrete actions NASA should take to advance GenAI capabilities and adoption. These recommendations span workforce development, infrastructure deployment, and strategic demonstrations, organized to address both immediate opportunities and longer-term capability building.

5.2.a NASA-Wide AI Exploration Event

Recommendation 1

Organize a comprehensive, agency-wide AI exploration event delivered as a hybrid program with virtual sessions accessible across all NASA centers supplemented by local in-person activities. The event would provide structured exposure to GenAI capabilities through demonstrations, hands-on workshops, and role-specific breakout sessions spanning engineering disciplines and business functions such as accounting, project management, contracts, and proposal writing. Employees select relevant sessions based on their roles and experience levels from multiple tracks ranging from introductory "what is GenAI" content for non-users to advanced workshops on prompt engineering, agent development, and domain-specific applications.

© 2026 California Institute of Technology. Government sponsorship acknowledged.

Time Horizon: Near-term (6-12 months). Initial planning and speaker coordination can begin immediately, with the event deliverable within one year of commitment.

Goals:

- Lower barrier to entry for employees who have not yet engaged with GenAI tools by providing safe, structured introduction to capabilities and appropriate use cases
- Inspire early adopters who are experimenting with tools but haven't discovered advanced applications relevant to their work
- Create cross-center connections between practitioners working on similar problems, reducing duplicative effort and accelerating knowledge transfer
- Demonstrate organizational commitment to GenAI adoption, providing explicit permission and encouragement for workforce to invest time learning these technologies

Context: Lawrence Livermore National Laboratory's aiEDGE for Innovation Day provides a proven model, having successfully engaged 3,200 employees across diverse job roles through demonstrations and breakout sessions focused on real-world applications [24]. Virtual sessions enable agency-wide participation and knowledge sharing while local activities foster center-specific communities of practice and allow hands-on experimentation with tools and workflows. Hybrid delivery accommodates NASA's geographically distributed workforce while maintaining the engagement benefits of in-person interaction.

Investment level: Moderate—requires dedicated event planning resources, speaker coordination across centers and external partners, infrastructure for hybrid delivery, and employee time allocation, but represents one-time investment with potential for annual recurrence.

© 2026 California Institute of Technology. Government sponsorship acknowledged.

5.2.b Public Benchmarks and Challenge Problems

Recommendation 2

Develop and publicly host NASA-specific datasets, benchmarks, and challenge problems that address gaps in public resources while meeting agency evaluation needs. Possible examples include

- **Database example**: database of non-ITAR engineering drawings and corresponding CAD to help facilitate the development of multimodal LLM capabilities
- **Benchmark example**: a set of NASA engineering and science multiple choice questions with corresponding answers to help evaluate the performance of LLMs on topics specific to NASA use cases
- **Design challenge example**: engineering design problems with relevant loading environments and requirements – similar to the GE jet engine bracket challenge or JPL's multiphysics optical mount design challenge – to aid in the development of Text-to-CAD and Text-to-Structure capabilities

To develop relevant and meaningful challenges, NASA should convene a workshop of domain experts and AI researchers to define challenge scope and categories, then establish infrastructure for public hosting with submission portals, automated evaluation, and leaderboard tracking.

Time Horizon: Near to mid-term (1-2 years). Initial challenge definition and infrastructure development can begin within months, with first challenges operational within 12-18 months and continuous expansion thereafter.

Goals:

- Engage broader research community including academia, industry, and citizen scientists in developing capabilities serving NASA needs, creating feature-building incentives that wouldn't emerge from commercial development alone
- Establish clear metrics for evaluating GenAI progress on NASA-relevant tasks, supporting trust-building through objective capability assessment rather than relying on vendor claims or limited internal evaluation
- Generate high-quality training datasets in aerospace domains where data scarcity currently limits capability development
- Provide public engagement opportunities that demonstrate NASA's technology leadership while addressing mission-relevant challenges

Context: Public benchmarks and challenges have proven to be significant contributors to AI progress over the past two decades, providing clear targets that focus research effort and enable objective capability assessment. ImageNet transformed computer vision research beginning in 2009 by providing annotated images (now at more than 14 million) to aid in ML object detection development and training and establishing annual challenges from 2010 through 2017 that allowed researchers to measure progress [44]. In engineering domains, similar approaches have

© 2026 California Institute of Technology. Government sponsorship acknowledged.

driven innovation: the GE jet engine bracket challenge on GrabCAD received 628 entries during the emergence of topology optimization in 2013 [45], while NASA's Common Research Model conceived in 2007 enabled aeronautical researchers across industry, government, and academia to work together on relevant computational fluid dynamics validation problems [46]. More recently in GenAI, domain-specific challenges like GPQA provide challenging multiple-choice questions in biology, chemistry, and physics authored by PhD-level experts, creating tools to evaluate progress on solving high-level expert problems [47]. Applied to NASA-specific technical challenges, this approach could accelerate GenAI capability development in aerospace domains where current foundation models underperform on specialized knowledge and reasoning tasks.

Investment level: Moderate—requires a funded workshop for challenge definition, ongoing infrastructure for challenge hosting and evaluation, prize funding to incentivize participation, and personnel time for benchmark curation and challenge administration. This approach could also be implemented internally through NASA Spark, potentially expanding existing programs rather than creating entirely new infrastructure.

5.2.c NASA Knowledge Source Integration

Recommendation 3

Develop and deploy MCPs providing programmatic access to critical institutional knowledge sources and research literature, enabling both institutional agent platforms and grassroots applications to leverage decades of accumulated technical expertise. Target both internal NASA knowledge repositories (such as lessons learned, mission documents, and technical repositories) and external research sources (journals, technical standards, reference books) through partnerships with publishers, standards organizations, and library systems.

Time Horizon: Near-term (1-2 years for substantial deployment). Initial MCP prototypes for high-priority knowledge sources can be developed within the first year, with progressive rollout across additional sources over the following 1-2 years as infrastructure matures.

Goals:

- Make institutional knowledge broadly accessible to GenAI applications without requiring custom integration work for each tool
- Reduce knowledge access time from hours spent searching disconnected systems to seconds through automated retrieval
- Enable grounding of LLM responses in authoritative NASA sources, addressing hallucination concerns through citation of original documents
- Preserve institutional memory by making historical lessons learned and design decisions discoverable when relevant to current problems

© 2026 California Institute of Technology. Government sponsorship acknowledged.

- Provide foundation for more sophisticated applications including analog mission search, requirements traceability checking, and design review automation

Context: NASA's historical knowledge spanning nearly seven decades represents one of the agency's greatest resources, but this knowledge remains locked in disconnected systems with inconsistent access methods that GenAI applications cannot easily query. Priority internal knowledge sources include:

- NASA Engineering and Safety Center (NESC) products: Lessons Learned database, Technical Bulletins, Engineering Reports, Technical Updates
- Materials and Processes Technical Information System (MAPTIS)
- NASA Scientific and Technical Information (STI) Repository
- Center-specific repositories: mission databases, technical wikis, design standards, heritage mission documentation

External research knowledge sources represent a critical gap in current GenAI infrastructure, as foundation models are trained only on public sources with knowledge cutoff dates that lag state-of-the-art research by months or years, requiring citeable integration through RAG systems rather than relying on training data alone. NASA should pursue partnerships to establish programmatic access to:

- Technical society publications: IEEE, AIAA, ASME
- Engineering standards: SAE, ASTM
- Technical reference works

MCPs provide standardized interfaces enabling LLM-based applications to query knowledge sources programmatically, retrieve relevant documents, and cite sources supporting generated responses. Unlike traditional APIs requiring custom integration code, MCPs follow conventions that LLMs can discover and use autonomously, dramatically reducing integration effort. Investment delivers value across all GenAI applications rather than requiring per-application integration efforts, creating multiplier effects as more applications leverage shared knowledge access infrastructure.

Investment level: Moderate to high—each knowledge source requires MCP development, security review, testing for performance at scale, and integration with authentication systems; ongoing maintenance addresses system updates and ensures continued compatibility.

© 2026 California Institute of Technology. Government sponsorship acknowledged.

5.2.d NASA-Wide Agent Platform

Recommendation 4

Mature and expand existing institutional agent platforms to build substantial user adoption and capability breadth across NASA. NASA and JPL have deployed initial agent platforms—ChatGSFC agency-wide and Onyx at JPL—but these remain early in maturity with limited shared agent libraries (approximately 20 agents on ChatGSFC and 36 on Onyx as of May 2026) and constrained functionality compared to mature implementations. Investment should focus on expanding platform capabilities through MCP integrations providing access to institutional knowledge sources, multimodal data handling, web search, and domain-specific tools; actively promoting platform adoption through demonstrations, training, and visible leadership encouragement; creating incentives for agent sharing and recognition of high-value contributions; and establishing communities of practice that accelerate knowledge transfer and capability development.

Time Horizon: Near-term (1-2 years for substantial capability expansion and user base growth). Capability expansion through MCP integration can proceed immediately in parallel with knowledge source development, while user adoption building requires sustained investment in training, promotion, and community development over the following 1-2 years.

Goals:

- Grow shared agent libraries from dozens to hundreds through active user engagement and contribution incentives
- Eliminate duplicative development effort by enabling visibility into existing capabilities and connecting practitioners working on similar problems
- Expand platform functionality beyond current limitations to support multimodal interactions, web search, and access to institutional knowledge sources
- Create data-driven understanding of which GenAI capabilities deliver the most value through usage metrics, informing future investment decisions
- Accelerate capability maturation through community feedback, testing, and improvement cycles

Context: NASA has established foundational agent platform infrastructure at both agency and center levels, but early adoption metrics reveal significant maturity gaps compared to industry implementations. Blue Origin's BlueGPT marketplace hosts more than 2,000 deployed agents, demonstrating the scale possible when platforms achieve broad organizational adoption [23]. The gap between NASA's current deployment (tens of agents) and mature implementations (thousands of agents) reflects not technical infrastructure deficiencies but rather limited capability breadth, insufficient user awareness, and absence of sustained organizational encouragement. Expanding platform capabilities depends substantially on the Knowledge Source Integration recommendation—institutional knowledge access through MCPs provides the foundation for high-value agents that drive user adoption.

© 2026 California Institute of Technology. Government sponsorship acknowledged.

Investment level: Moderate—requires build out upon existing platform infrastructure, ongoing operations and maintenance, security review and accreditation for handling sensitive data, user training and support, and governance processes for agent approval and quality assurance.

5.2.e Dedicated Flight Project GenAI Consultant

Recommendation 5

Establish a funded position within select flight projects specifically dedicated to identifying, prototyping, and integrating GenAI solutions tailored to mission-specific workflows and challenges. This person would work independently while participating in relevant project meetings and reviews, developing hands-on understanding of project demands to identify GenAI application opportunities that may not be obvious to practitioners focused on immediate execution. The position requires someone familiar with both spacecraft project development and GenAI capabilities, funded at minimum half-time but ideally full-time. Responsibilities include prototyping promising GenAI solutions, pitching validated approaches to project teams, and supporting integration when teams decide to adopt the capabilities.

Time Horizon: Immediate (can begin on currently forming missions or add to select ongoing projects). Position implementation can proceed as soon as suitable candidates are identified, with value delivery beginning within months as the explorer gains project context and begins prototyping applications.

Goals:

- Provide dedicated attention to GenAI opportunity identification that doesn't compete with immediate project execution pressures
- Enable evaluation of GenAI solutions within authentic project contexts where applicability and value become clear through hands-on prototyping
- Reduce adoption friction by having someone who understands both the project's operational realities and GenAI capabilities bridge the gap between potential and implementation
- Demonstrate organizational commitment to GenAI exploration through funded positions rather than relying on discretionary efforts
- Generate project-specific knowledge about which GenAI applications deliver value in real mission contexts rather than theoretical assessments

Context: Many potentially valuable GenAI applications remain undiscovered because practitioners immersed in immediate project demands lack time to experiment with emerging capabilities, while AI researchers lack the mission context to identify where tools would provide genuine value. This disconnect means promising applications that could streamline workflows or improve quality go unrecognized until after projects complete. A dedicated position addresses this gap by combining spacecraft project knowledge with GenAI expertise in someone whose explicit respon-

© 2026 California Institute of Technology. Government sponsorship acknowledged.

sibility is discovering opportunities rather than executing current tasks. The position resembles innovation roles common in industry, where dedicated resources explore emerging technologies and prototype applications before broader organizational adoption, but focused specifically on GenAI capabilities and applications. Success depends on two critical factors: first, the person must have credibility with engineering teams through direct spacecraft project experience rather than coming from a pure AI background; second, funding must provide protected time rather than expecting exploration to happen alongside full-time project responsibilities. Implementation could occur through multiple organizational models: positions established on select high-priority missions, rotational assignments where experienced engineers spend 6-12 months in an exploration role before returning to traditional project work, or center-level positions supporting multiple smaller missions.

Investment level: Low to moderate per position—requires salary funding for minimum half-time allocation (ideally full-time), computing resources for prototyping and experimentation, and discretionary budget for tools and training, but represents straightforward personnel investment with immediate value potential.

5.2.f Large-Scale Demonstration Project

Recommendation 6

Fund a large-scale demonstration project that applies GenAI-augmented workflows to design and build flight hardware for an actual space mission (such as a smallsat technology demonstration mission), substantially exceeding the scope of individual proofs-of-concept. Select a flight project with meaningful hardware complexity—spacecraft bus, instrument, or subsystem—where design, analysis, fabrication, and testing occur within the demonstration timeframe. Integrate GenAI capabilities across multiple engineering disciplines and include non-engineering workflows such as requirements management, documentation, procurement, and project planning. Establish clear baseline comparison enabling measurement of cycle time reduction, cost savings, and design iteration velocity. Key deliverables include comprehensive lessons learned documenting successful and failed approaches, required organizational changes, process modifications for GenAI integration, verification and validation approaches for AI-generated outputs, and quantitative metrics on productivity improvement.

Time Horizon: Future opportunity requiring annual reassessment of software ecosystem maturity. Current GenAI tools lack sufficient spatial awareness capability for large-scale autonomous design workflows, making this demonstration premature today, but NASA should reassess readiness annually as the software ecosystem matures to identify the appropriate timing for initiating this multi-year demonstration.

Goals:

© 2026 California Institute of Technology. Government sponsorship acknowledged.

- Provide existence proof that GenAI-augmented workflows can successfully deliver flight hardware meeting NASA standards, addressing skepticism about technology readiness
- Identify specific organizational changes, management support requirements, and resource allocations necessary for large-scale deployment—information that cannot be derived from small-scale pilots or individual tool demonstrations
- Generate institutional knowledge about effective human-AI collaboration patterns in aerospace engineering, creating foundation for broader deployment
- Demonstrate leadership commitment to GenAI adoption, providing visible signal that encourages workforce investment in capability development

Context: The pilot study conducted for this report provided valuable initial insights into GenAI capabilities and workflow patterns, but lacked sufficient scope to fully implement and understand this new design paradigm—particularly the organizational structures, processes, and top-down support required for successful large-scale deployment. This demonstration targets the engineering process itself rather than demonstrating novel hardware capabilities—the objective is not to prove that GenAI can design better brackets or optimize structural performance (those capabilities are established through existing proofs-of-concept), but rather to reveal what project-level organization, institutional support structures, process modifications, and cultural changes enable teams to actually realize GenAI's potential at mission scale. Understanding these organizational factors matters more than hardware performance improvements because organizational barriers currently constrain deployment more than technical limitations. Like a technology demonstration but for the engineering process, this project would rigorously document what works, what fails, and what organizational conditions enable success.

Investment level: High—requires dedicated project funding at the scale of a technology demonstration mission, management attention and organizational change expertise, and willingness to accept schedule risk if GenAI approaches encounter unexpected challenges; however, delivers knowledge and organizational learning that cannot be obtained through smaller investments.

These recommendations collectively address workforce development (NASA AI exploration event), capability development (benchmarks, agent platform, knowledge integration), and organizational learning (large-scale demonstration). Implementation should proceed in parallel where possible, with infrastructure investments (agent platform, knowledge MCPs) enabling grassroots innovation while strategic demonstrations (large-scale project) inform policy and organizational evolution.

© 2026 California Institute of Technology. Government sponsorship acknowledged.

6 Conclusions

This NASA Planetary Exploration Science Technology Office (PESTO) study provides systematic assessment of generative AI capabilities for spacecraft design and development, cutting through extraordinary hype to distinguish genuine capabilities from aspirational claims. Our findings synthesize industry engagement across NASA centers and aerospace companies, hands-on pilot demonstrations applying GenAI tools to representative design problems, and evaluation of organizational factors affecting adoption. The assessment reveals a technology landscape where immediate deployment opportunities coexist with fundamental limitations requiring continued development.

Application maturity varies dramatically across the GenAI landscape for spacecraft design, revealing clear distinctions between capabilities meeting expectations and those falling short of aspirational claims. Contained low-level tasks—code generation, knowledge retrieval, interactive design dashboards, and requirements derivation from heritage missions—are ready for deployment with appropriate human validation, though even these mature capabilities face practical constraints including persistent hallucination and substantial verification burden imposing real cognitive load on practitioners. In contrast, high-level tasks like autonomous design of subsystems or complete spacecraft remain nascent, requiring fundamental breakthroughs in spatial reasoning and physics-informed generation before practical utility.

In conjunction with technical challenges, organizational readiness presents equally significant barriers to adoption—in many cases constituting the larger constraint than technology maturity itself. Infrastructure access improves steadily with secure cloud deployments and institutional platforms emerging, yet knowledge integration efforts remain underdeveloped and agent platforms exist only in limited deployments. Workforce development lags critically behind tool availability: high barriers to entry persist, training programs remain minimal, and wide skill gaps continue despite improving access. Strategic deployment planning remains limited, with grassroots experimentation dominating while coordinated organizational capability building remains rare. These organizational factors—infrastructure gaps, workforce skill development needs, institutional culture, and strategic planning limitations—must be addressed alongside technical maturation for NASA to realize GenAI's transformational potential.

NASA's path forward requires balancing near-term deployment of high-maturity tools with strategic investments that position the agency for long-term capability growth. The overarching investment principle is clear: leverage commercial foundation models as computational engines that continue improving through industry competition rather than duplicating commercial development. NASA should concentrate investment on capabilities the broader AI ecosystem cannot provide or lacks sufficient market incentive to develop—integrating institutional knowledge through Model Context Protocols that connect decades of mission data and technical expertise; developing aerospace-specific benchmarks and challenge problems that focus research community attention on NASA-relevant technical challenges; training the workforce on domain-specific applications through sustained programs rather than one-time events; and demonstrating through large-scale projects what organizational factors enable successful deployment at mission scale. This approach captures immediate value from mature tools like coding assistants and knowledge copilots while building the infrastructure, workforce capabilities, and organizational learning required to leverage emerging capabilities as they achieve production readiness. The window for strategic action during this technology inflection point will not remain open indefinitely—organizations that build expertise and infrastructure now will compound advantages as capabilities mature, while those that delay face widening gaps and steeper adoption curves when competitive pressure eventually forces reactive deployment.

© 2026 California Institute of Technology. Government sponsorship acknowledged.

The transformative potential extends beyond productivity gains to fundamentally reshaping how NASA conducts spacecraft development. Engineers augmented with conversational access to decades of institutional knowledge can make better-informed decisions faster, learning from heritage missions without years spent building domain expertise. Design iteration cycles that once required weeks compress to days, enabling exploration of broader trade spaces and more ambitious mission concepts within equivalent development timelines. Crucially, the design intent and engineering rationale that historically evaporated after project completion now gets captured systematically, creating institutional memory that compounds across missions and makes hard-won lessons genuinely reusable rather than repeatedly rediscovered. NASA stands uniquely positioned not merely to adopt these tools but to lead the aerospace community in establishing best practices for responsible GenAI deployment in safety-critical engineering—demonstrating through rigorous application how to validate AI-generated outputs, when to trust automated reasoning, and how to maintain the technical excellence that enables exploration of increasingly distant and challenging destinations. The path forward demands sustained commitment and thoughtful execution, but the destination is clear: an aerospace engineering practice where human creativity and judgment are amplified by computational capabilities that make institutional knowledge conversationally accessible, design exploration dramatically more efficient, and the next generation of space missions simultaneously more capable and more achievable.

7 Acknowledgements

This study was sponsored by the NASA Planetary Exploration Science Technology Office (PESTO), whose support enabled systematic assessment of GenAI capabilities for spacecraft design and development. The research was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration (80NM0018D0004).

We thank the expert practitioners who generously shared their time, insights, and organizational experiences through interviews and symposium participation. From NASA centers, we engaged with experts at Jet Propulsion Laboratory, Goddard Space Flight Center, Marshall Space Flight Center, Ames Research Center, Langley Research Center, and Johnson Space Center. From the aerospace industry and national laboratories, we benefited from discussions with iMETALX, Motiv Space Systems, Xometry, and Lawrence Livermore National Laboratory. From the software ecosystem, we acknowledge valuable engagement with both AI-native companies—Celedon, P1 AI, Mecado, Pensa Technologies, and Zoo—and incumbent software companies integrating GenAI capabilities—Autodesk, InfinitForm, nTop, Synera, XMPro, NVIDIA, and ANSYS.

We are grateful to Ben Nuernberger (JPL), Elie Cuevas (Mecado), and Antti Pulkkinen (Goddard), Luc Peterson (Lawrence Livermore National Laboratory), and Ryan McClelland (Goddard) for their thorough review of draft versions of this report and their constructive feedback that substantially improved the final document.

Bibliography

[1] M. Suleyman, "Mustafa Suleyman, Microsoft AI chief executive, on the future of artificial intelligence." [Online]. Available: https://www.youtube.com/watch?v=YTrBz6Z5c0E

[2] M. Shumer, "Something Big is Happening." [Online]. Available: https://shumer.dev/something-big-is-happening

© 2026 California Institute of Technology. Government sponsorship acknowledged.

[3] NASA, "SEH 3.0 NASA Program/Project Life Cycle." [Online]. Available: https://www.nasa.gov/reference/3-0-nasa-program-project-life-cycle/

[4] K. Sharifani and M. Amini, "Machine Learning and Deep Learning: A Review of Methods and Applications." Accessed: Apr. 08, 2026. [Online]. Available: https://papers.ssrn.com/abstract=4458723

[5] D. Dhall, R. Kaur, and M. Juneja, "Machine Learning: A Review of the Algorithms and Its Applications," in *Proceedings of ICRIC 2019*, P. K. Singh, A. K. Kar, Y. Singh, M. H. Kolekar, and S. Tanwar, Eds., Cham: Springer International Publishing, 2020, pp. 47–63. doi: 10.1007/978-3-030-29407-6_5.

[6] N. Abcouwer *et al.*, "Machine Learning Based Path Planning for Improved Rover Navigation," in *2021 IEEE Aerospace Conference (50100)*, Big Sky, MT, USA: IEEE, Mar. 2021, pp. 1–9. doi: 10.1109/AERO50100.2021.9438337.

[7] M. Ono *et al.*, "MAARS: Machine learning-based Analytics for Automated Rover Systems," in *2020 IEEE Aerospace Conference*, Big Sky, MT, USA: IEEE, Mar. 2020, pp. 1–17. doi: 10.1109/AERO47225.2020.9172271.

[8] E. Goh, H. S. Venkataram, M. Hoffmann, M. D. Johnston, and B. Wilson, "Scheduling the NASA Deep Space Network with Deep Reinforcement Learning," in *2021 IEEE Aerospace Conference (50100)*, Big Sky, MT, USA: IEEE, Mar. 2021, pp. 1–10. doi: 10.1109/AERO50100.2021.9438519.

[9] A. ALDabbas, Z. Mustafa, and Z. Gal, "Advanced Trajectory Analysis of NASA's Juno Mission Using Unsupervised Machine Learning: Insights into Jupiter's Orbital Dynamics," *Future Internet*, vol. 17, no. 3, p. 125, Mar. 2025, doi: 10.3390/fi17030125.

[10] B. Rothrock, R. Kennedy, C. Cunningham, J. Papon, M. Heverly, and M. Ono, "SPOC: Deep Learning-based Terrain Classification for Mars Rover Missions," in *AIAA SPACE 2016*, Long Beach, California: American Institute of Aeronautics, Astronautics, Sept. 2016. doi: 10.2514/6.2016-5539.

[11] V. Vishwarupe *et al.*, "Comparative Analysis of Machine Learning Algorithms for Analyzing NASA Kepler Mission Data," *Procedia Computer Science*, vol. 204, pp. 945–951, Jan. 2022, doi: 10.1016/j.procs.2022.08.115.

[12] A. E. Stott *et al.*, "Machine learning and marsquakes: a tool to predict atmospheric-seismic noise for the NASA InSight mission," *Geophysical Journal International*, vol. 233, no. 2, pp. 978–998, May 2023, doi: 10.1093/gji/ggac464.

[13] D. Dinesh, S. Kumar, and S. Saran, "Machine Learning Modelling for Soil Moisture Retrieval from Simulated NASA-ISRO SAR (NISAR) L-Band Data," *Remote Sensing*, vol. 16, no. 18, p. 3539, Jan. 2024, doi: 10.3390/rs16183539.

[14] S. Adeli, B. Salehi, M. Mahdianpari, L. J. Quackenbush, and B. Chapman, "Moving Toward L-Band NASA-ISRO SAR Mission (NISAR) Dense Time Series: Multipolarization Object-Based Classification of Wetlands Using Two Machine Learning Algorithms," *Earth and Space Science*, vol. 8, no. 11, p. e2021EA001742, Nov. 2021, doi: 10.1029/2021EA001742.

[15] C. A. Nixon *et al.*, "Feature Extraction and Classification from Planetary Science Datasets enabled by Machine Learning," in *2023 IEEE Aerospace Conference*, Big Sky, MT, USA: IEEE, Mar. 2023, pp. 1–16. doi: 10.1109/AERO55745.2023.10115556.

[16] L. Banh and G. Strobel, "Generative artificial intelligence," *Electronic Markets*, vol. 33, no. 1, p. 63, Dec. 2023, doi: 10.1007/s12525-023-00680-1.

© 2026 California Institute of Technology. Government sponsorship acknowledged.

[17] A. Vaswani *et al.*, "Attention is All you Need," Long Beach, California, 2017.

[18] Anthropic, "Reasoning models don't always say what they think." [Online]. Available: https://www.anthropic.com/research/reasoning-models-dont-say-think

[19] Y. Gao *et al.*, "Retrieval-Augmented Generation for Large Language Models: A Survey." Accessed: May 19, 2026. [Online]. Available: http://arxiv.org/abs/2312.10997

[20] S. Yao *et al.*, "ReAct: Synergizing Reasoning and Acting in Language Models." Accessed: May 20, 2026. [Online]. Available: http://arxiv.org/abs/2210.03629

[21] Anthropic, "Model Context Protocol." [Online]. Available: https://modelcontextprotocol.io/

[22] S. Allmendinger, L. Bonenberger, K. Endres, D. Fetzer, H. Gimpel, and N. Kühl, "Multi-agent AI," *Electronic Markets*, vol. 36, no. 1, p. 18, Feb. 2026, doi: 10.1007/s12525-025-00862-z.

[23] Amazon Web Services, "Blue Origin Uses AWS to Democratize Access to Generative AI and Power Innovation." Accessed: Apr. 08, 2026. [Online]. Available: https://aws.amazon.com/solutions/case-studies/blue-origin-case-study/

[24] Lawrence Livermore National Laboratory, "LLNL employees dive into AI's transformative potential at aiEDGE for Innovation Day." Accessed: Apr. 08, 2026. [Online]. Available: https://www.llnl.gov/article/52701/llnl-employees-dive-ais-transformative-potential-aiedge-innovation-day

[25] U.S. Department of Energy, "The Genesis Mission." Accessed: Apr. 08, 2026. [Online]. Available: https://www.energy.gov/undersecretaryforscience/genesis-mission/genesis-mission

[26] M. Ataei, F. Askari, K. R. Malekshan, and P. K. Jayaraman, "Zero-to-CAD: Agentic Synthesis of Interpretable CAD Programs at Million-Scale Without Real Data." Accessed: May 18, 2026. [Online]. Available: http://arxiv.org/abs/2604.24479

[27] R. Royce *et al.*, "Enabling Novel Mission Operations and Interactions with ROSA: The Robot Operating System Agent." Accessed: Apr. 23, 2026. [Online]. Available: http://arxiv.org/abs/2410.06472

[28] B. Nuernberger, A. Liu, H. Stefanini, R. Otis, A. Towler, and R. P. Dillon, "AiBAT: Artificial Intelligence/Instructions for Build, Assembly, and Test," in *2025 IEEE 11th International Conference on Space Mission Challenges for Information Technology (SMC-IT)*, July 2025, pp. 81–89. doi: 10.1109/SMC-IT64853.2025.00017.

[29] R. McClelland, "From Text to Spaceship: Advancing AI in Aerospace." [Online]. Available: https://cdfam.com/nasa-from-text-to-spaceship-advancing-ai-in-aerospace/

[30] M. Yavartanoo, S. Hong, R. Neshatavar, and K. M. Lee, "Text2CAD: Text to 3D CAD Generation via Technical Drawings." Accessed: May 08, 2026. [Online]. Available: http://arxiv.org/abs/2411.06206

[31] Y. Guan, X. Wang, X. Xing, J. Zhang, D. Xu, and Q. Yu, "CAD-Coder: Text-to-CAD Generation with Chain-of-Thought and Geometric Reward," 2025.

[32] B. K. Muirhead, A. Nicholas, and J. Umland, "Mars Sample Return Mission Concept Status," in *2020 IEEE Aerospace Conference*, Mar. 2020, pp. 1–8. doi: 10.1109/AERO47225.2020.9172609.

[33] K. A. Farley *et al.*, "Mars 2020 Mission Overview," *Space Science Reviews*, vol. 216, no. 8, p. 142, Dec. 2020, doi: 10.1007/s11214-020-00762-y.

© 2026 California Institute of Technology. Government sponsorship acknowledged.

[34] F. Mier-Hicks *et al.*, "Sample Recovery Helicopter," in *2023 IEEE Aerospace Conference*, Mar. 2023, pp. 1–11. doi: 10.1109/AERO55745.2023.10115951.

[35] J. Balaram, M. Aung, and M. P. Golombek, "The Ingenuity Helicopter on the Perseverance Rover," *Space Science Reviews*, vol. 217, no. 4, p. 56, June 2021, doi: 10.1007/s11214-021-00815-w.

[36] W. J. F. Koning, W. Johnson, and B. G. Allan, "Generation of Mars Helicopter Rotor Model for Comprehensive Analyses," San Francisco, California, Jan. 2018.

[37] S. Cappucci and M. T. Pauken, "Thermal performance of Ingenuity, the Mars Helicopter," July 2022.

[38] T. Tzanetos *et al.*, "Ingenuity Mars Helicopter: From Technology Demonstration to Extraterrestrial Scout," in *2022 IEEE Aerospace Conference (AERO)*, Big Sky, MT, USA: IEEE, Mar. 2022, pp. 1–19. doi: 10.1109/AERO53065.2022.9843428.

[39] S. A. Langberg, J. D. Tyler, B. T. Pipenberg, M. T. Keennon, and P. Zwaan, "Landing Gear Design, Fabrication, and Testing for the Ingenuity Mars Helicopter," in *2022 IEEE Aerospace Conference (AERO)*, Big Sky, MT, USA: IEEE, Mar. 2022, pp. 1–13. doi: 10.1109/AERO53065.2022.9843527.

[40] C. Dull, L. Wagner, L. Young, and W. Johnson, "Hover and Forward Flight Performance Modeling of the Ingenuity Mars Helicopter," San Jose, CA, Jan. 2022.

[41] S. Withrow-Maser *et al.*, "Mars Sample Recovery Helicopter: Rotorcraft to Retrieve the First Samples from the Martian Surface," in *Proceedings of the Vertical Flight Society 79th Annual Forum and Technology Display*, West Palm Beach, Florida: The Vertical Flight Society, May 2023, pp. 1–8. doi: 10.4050/F-0079-2023-17969.

[42] W. J. F. Koning and M. Dominguez, "Mars Helicopter Ingenuity Rotor Geometry," NASA STI Program Report Series San Francisco, California, 2024.

[43] H. F. Grip *et al.*, "The Chopper Next-Generation Mars Rotorcraft: Scaling Ingenuity by a Factor 20," in *2025 IEEE Aerospace Conference*, Big Sky, MT, USA: IEEE, Mar. 2025, pp. 1–12. doi: 10.1109/AERO63441.2025.11068461.

[44] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, June 2009, pp. 248–255. doi: 10.1109/CVPR.2009.5206848.

[45] GrabCAD, "GE Jet Engine Bracket Challenge." [Online]. Available: https://grabcad.com/challenges/ge-jet-engine-bracket-challenge

[46] M. B. Rivers, "NASA Common Research Model: A History and Future Plans," Reston, VA, United States, Jan. 2019. doi: 10.2514/6.2019-2188.

[47] D. Rein *et al.*, "GPQA: A Graduate-Level Google-Proof Q&A Benchmark." Accessed: May 08, 2026. [Online]. Available: http://arxiv.org/abs/2311.12022

© 2026 California Institute of Technology. Government sponsorship acknowledged.