Spatial Intelligence at Scale

AtlasPro AI's Approach to Building Agentic Geospatial Systems

Technical Report

Version 3.0 | January 2026

Gloria Felicia — Research Lead
Nolan Bryant — Systems Architect
Handi Putra — ML Engineer
Ayaan Gazali — Research Engineer
Eliel Lobo — Research Engineer
Esteban Rojas — Research Engineer

AtlasPro AI

Research Division

Correspondence: research@atlaspro.ai

Abstract

This technical report presents AtlasPro AI's comprehensive research approach to building autonomous spatial intelligence systems for critical infrastructure in the telecommunications and utilities sectors. We introduce a unified three-axis taxonomy (Task × Capability × Scale) that organizes the intersection of agentic AI capabilities with spatial task domains across multiple operational scales. Our preliminary research synthesizes findings from over 800 peer-reviewed papers from top-tier venues including NeurIPS, ICML, ICLR, CVPR, CoRL, and RSS.

Our analysis reveals critical gaps in existing approaches that AtlasPro AI is uniquely positioned to address. Current systems excel within narrow operational envelopes but fail systematically when tasks require cross-scale reasoning or long-horizon planning under geometric constraints. We identify six systematic failure modes that plague existing spatial AI systems: spatial hallucination, scale confusion, temporal incoherence, constraint violation, compositional failure, and distribution shift fragility. These failures are particularly acute in the network infrastructure domain, where no existing solution combines leading AI agent technology with deep vertical expertise.

We present a detailed competitive analysis of over 40 companies across six categories, demonstrating a significant market opportunity at the intersection of agentic AI and network intelligence. Our analysis confirms that AtlasPro AI is creating a new category: no incumbent GIS platform has AI-native architecture, and no AI platform company has the domain-specific expertise for network topology reasoning.

This report documents our research methodology, presents the three-axis taxonomy as a framework for system design, provides comprehensive technical deep-dives into core components (GNNs, World Models, VLAs, MCP), details AtlasPro AI's architectural principles, and outlines our three-phase research roadmap through 2027. We release this report to establish priority on our methodological contributions and to invite collaboration from the research community.

Keywords: Spatial Intelligence, Agentic AI, World Models, Graph Neural Networks, Geospatial AI, Network Intelligence, Telecom, Utilities, Model Context Protocol.

Contents

Part I
Introduction and Vision

1 The Spatial Intelligence Imperative

Large language models have achieved remarkable success in symbolic reasoning, code generation, and natural language understanding [Brown et al., 2020, OpenAI, 2023, Touvron et al., 2023, Team and Google, 2023, Anthropic, 2024]. Yet these same models fail systematically when confronted with the physical world. Navigation agents hallucinate paths through walls. Manipulation planners propose grasps that violate basic physics. Embodied systems misjudge distances by orders of magnitude [Chen et al., 2024a, Yang et al., 2025]. The gap between linguistic competence and spatial competence represents one of the most significant barriers to deploying AI systems in real-world applications, particularly within critical infrastructure sectors like telecommunications and utilities where precision and reliability are paramount.

AtlasPro AI was founded to bridge this gap. Our research program investigates how to build autonomous systems that can perceive three-dimensional structure, reason about object relationships under physical constraints, and execute actions that respect the geometry of the world. This is not merely an incremental improvement over language understanding; it requires fundamentally different representations, architectures, and training paradigms tailored to the unique challenges of network infrastructure.

1.1 Why Spatial Intelligence Matters Now

Three converging trends make spatial intelligence tractable and urgent:

Foundation Model Capabilities. Large language models now exhibit emergent reasoning capabilities [Wei et al., 2022]. Vision-language models can understand complex scenes [Liu et al., 2023b, Alayrac et al., 2022]. The question is no longer whether AI can reason, but whether it can reason about the physical world.

Robotics at Scale. Open-source robotics datasets and foundation models have democratized embodied AI research [Team et al., 2024, Kim et al., 2024]. The barrier to entry has dropped dramatically, enabling rapid iteration on spatial AI systems.

Industry Demand. Autonomous vehicles, warehouse robotics, drone delivery, and smart city infrastructure all require spatial intelligence. The market opportunity exceeds $100 billion by 2030. More specifically, the demand for intelligent planning and management of critical infrastructure like fiber optic and utility networks represents a multi-billion dollar market ripe for disruption.

AtlasPro AI is positioned at this intersection. Our research program aims to develop the foundational capabilities for spatially-aware autonomous systems, with an initial focus on the telecommunications and utilities sectors where our team has deep domain expertise.

1.2 Scope and Limitations of This Report

This report presents AtlasPro AI's research methodology, competitive analysis, and preliminary findings. It does not describe a deployed system or report experimental results from a novel architecture. We are transparent about what this report is and is not:

What This Report Is:

- A comprehensive literature synthesis of over 800 papers

- A unified taxonomy for organizing the spatial AI design space
- A detailed competitive landscape analysis
- A set of architectural principles derived from our analysis
- A research roadmap for AtlasPro AI's development program

What This Report Is Not:

- A peer-reviewed publication
- A description of a deployed production system
- A report of novel experimental results
- A product specification or engineering design document

We release this report to establish priority on our methodological contributions and to invite feedback from the research community.

1.3 Document Structure

This technical report is organized into six parts:

Part I: Introduction and Vision establishes the strategic context, defines key concepts, and presents our unified three-axis taxonomy.

Part II: Competitive Landscape provides a comprehensive analysis of over 40 companies, demonstrating AtlasPro AI's unique market position.

Part III: Technical Foundations presents deep-dives into core technical components: agentic architectures, memory systems, planning, and tool use.

Part IV: Enabling Technologies covers GNNs, world models, vision-language-action models, and geospatial foundation models.

Part V: AtlasPro AI's Approach details our architectural principles, MCP integration strategy, and differentiated technical approach.

Part VI: Research Roadmap and Conclusion outlines our three-phase development plan and identifies grand challenges for the field.

2 Foundational Concepts and Definitions

2.1 Defining Agentic AI

We adopt the definition from Wang et al. [2024b]: an AI agent is an autonomous entity that perceives its environment, makes decisions, and takes actions to achieve specific goals. This definition encompasses three core capabilities that form our taxonomy's Capability axis:

Memory. The ability to accumulate and retrieve knowledge across time. For spatial agents, this includes both episodic memory (what happened where) and semantic memory (general spatial knowledge). Memory systems range from short-term context windows to long-term retrieval-augmented generation [Packer et al., 2023, Lewis et al., 2020].

Planning. The ability to decompose goals into executable action sequences. Planning under geometric constraints requires hybrid approaches that combine the flexibility of neural models with the rigor of symbolic methods [Garrett et al., 2021, Silver et al., 2024].

Tool Use. The ability to extend capabilities through external tools and APIs. For spatial agents, this includes perception APIs, robot control interfaces, and GIS tools [Schick et al., 2023, Patil et al., 2023].

These agents operate through iterative cycles of perception, reasoning, action, and feedback [Yao et al., 2023, Shinn et al., 2023].

2.2 Defining Spatial Intelligence

We define Spatial Intelligence as the ability to perceive 3D structure, reason about object relationships, navigate environments, and manipulate physical objects [Chen et al., 2024a, Marr, 1982]. This encompasses four primary task domains that form our taxonomy's Task axis:

Navigation. Moving through environments toward goals. The core challenge is grounding linguistic instructions in traversable paths while avoiding obstacles and respecting physical constraints [Anderson et al., 2018, Batra et al., 2020].

Scene Understanding. Perceiving and representing 3D structure. The core challenge is building representations that support downstream reasoning, including object detection, semantic segmentation, and relationship inference [Dai et al., 2017, Krishna et al., 2017].

Manipulation. Interacting with objects through physical contact. The core challenge is planning contact-rich interactions under uncertainty, including grasping, placement, and tool use [Zeng et al., 2021, Shridhar et al., 2022].

Geospatial Analysis. Reasoning about large-scale spatial phenomena. The core challenge is handling heterogeneous data sources at city-to-global scales, including satellite imagery, sensor networks, and infrastructure graphs [Jakubik et al., 2024, Mai et al., 2023]. This is the primary focus of AtlasPro AI.

2.3 The Three-Axis Taxonomy

We propose a unified taxonomy that organizes the intersection of agentic AI and spatial intelligence. The taxonomy comprises three orthogonal axes:

Axis 1: Spatial Task. Navigation, Scene Understanding, Manipulation, Geospatial Analysis.

Axis 2: Agentic Capability. Memory, Planning, Tool Use.

Axis 3: Spatial Scale. Micro-spatial (<1m), Meso-spatial (1m–100m), Macro-spatial (>100m).

Key Insight: Scale Determines Architecture

Methods optimized for one scale often fail at others. A unified spatial AI system must bridge these scales, which remains an open challenge. AtlasPro AI focuses primarily on the macro-spatial scale, where GNNs provide a significant advantage in modeling network topologies, while maintaining awareness of cross-scale requirements for complete solutions.

3 Research Methodology

3.1 Literature Review Process

This report follows a systematic literature review methodology consistent with best practices in computer science [Kitchenham, 2004, Petersen et al., 2008]. We queried complementary academic databases: Google Scholar for breadth, arXiv for recent preprints, ACM Digital Library and IEEE Xplore for peer-reviewed systems research, Semantic Scholar for citation-aware ranking, and DBLP for comprehensive venue coverage.

Table 1: Three-Axis Taxonomy: Representative Methods Mapped to Axes

Method	Task	Capability	Scale	Key Innovation
VLMaps [Huang et al., 2023a]	Navigation	Memory	Meso	Language-indexed spatial maps
SayCan [Ahn et al., 2022]	Manipulation	Planning	Micro	Affordance-grounded LLM planning
RT-2 [Brohan et al., 2023]	Manipulation	Tool Use	Micro	Vision-language-action model
DreamerV3 [Hafner et al., 2023]	All	Planning	All	Universal world model
Prithvi [Jakubik et al., 2024]	Geospatial	Memory	Macro	Geospatial foundation model
DCRNN [Li et al., 2018]	Geospatial	Memory	Macro	Spatio-temporal GNN

Our search keywords included: "agentic AI," "spatial intelligence," "embodied AI," "vision-language navigation," "robot manipulation," "geospatial AI," "world models," "graph neural networks," "spatio-temporal learning," "vision-language-action," "foundation models for robotics," "telecom network optimization," and "utility infrastructure AI." Our initial search yielded over 3,000 papers.

3.2 Filtering Criteria

We applied a rigorous multi-stage filtering process:

Temporal Filtering. We selected papers published between 2018 and 2026, with emphasis on recent advances while including foundational works that established key paradigms.

Venue Filtering. We prioritized papers from top-tier venues including NeurIPS, ICML, ICLR, CVPR, ECCV, ICCV, CoRL, RSS, IROS, ICRA, ACM Computing Surveys, IEEE TPAMI, Nature, Science, and Science Robotics.

Quality Filtering. We prioritized papers with high citation counts and foundational methods, while explicitly including recent low-citation works that introduce paradigm-shifting approaches to avoid recency bias.

Relevance Filtering. We ensured papers directly addressed the intersection of agentic capabilities and spatial intelligence.

This process resulted in a final corpus of over 800 papers, which were systematically analyzed to derive the taxonomy, identify key trends, and synthesize the findings presented in this report.

Part II
Competitive Landscape and Differentiation

4 Market Landscape Overview

The market for geospatial technology is mature and dominated by large incumbents. The AI-powered geospatial intelligence market is a more recent, high-growth area with well-funded startups. However, our analysis reveals a critical gap: no company is effectively combining leading AI agent technology (LLMs, GNNs) with a deep vertical focus on critical network infrastructure (telecom, utilities).

Incumbents are retrofitting AI onto legacy GIS platforms, resulting in clunky, inefficient workflows. AI startups are building horizontal platforms or focusing on satellite/aerial imagery, not the underlying network graph. This leaves a significant, addressable market for a company that is both AI-native and vertically-focused. AtlasPro AI is designed to fill this gap.

4.1 Market Size and Growth

The global geospatial analytics market was valued at approximately $75 billion in 2023 and is projected to reach $150 billion by 2030, representing a compound annual growth rate (CAGR) of approximately 10%. The AI-powered segment is growing faster, with projections suggesting a CAGR of 25-30%.

Within this market, the telecommunications network planning and management segment represents approximately $5-8 billion annually, with utilities adding another $3-5 billion. These segments are characterized by:

- High-value, mission-critical applications
- Complex, graph-structured data
- Strong demand for predictive analytics
- Limited AI adoption to date
- High switching costs for incumbent solutions

5 Competitive Analysis by Category

Our research analyzed over 40 companies across six categories. The following sections summarize our findings.

5.1 Category 1: Traditional GIS Platforms

Analysis. Traditional GIS platforms like Esri's ArcGIS are powerful but fundamentally tool-based. Users must manually configure analyses, interpret results, and make decisions. These platforms are adding AI features (e.g., Esri's GeoAI), but the architecture is retrofitted rather than AI-native. AtlasPro AI's agentic approach enables autonomous analysis and decision-making, a paradigm shift from human-in-the-loop GIS.

Table 2: Traditional GIS Platform Competitors

Company	Funding	Overlap	AtlasPro Differentiation
Esri	Public	Indirect (4/10)	AI-native architecture; Agentic workflows vs. tool-based GIS
CARTO	$80M	Indirect (3/10)	Vertical focus; GNN-based reasoning vs. SQL analytics
Mapbox	$280M	Indirect (2/10)	Infrastructure focus vs. consumer mapping
Hexagon	Public	Indirect (3/10)	AI-first approach vs. CAD/GIS legacy

Table 3: Telecom Network Planning Competitors

Company	Funding	Overlap	AtlasPro Differentiation
IQGeo (Comsof)	Public	Direct (8/10)	GNN-based predictive optimization vs. heuristic automation
3-GIS	Acquired	Direct (7/10)	AI-driven insights vs. manual data management
VertiGIS	Acquired	Direct (6/10)	Predictive analytics vs. network documentation
Bentley	Public	Partial (5/10)	Agentic workflows vs. engineering design tools
GeoTel	Private	Adjacent (4/10)	Active intelligence vs. passive data provision

5.2 Category 2: Telecom Network Planning

Analysis. This is AtlasPro AI's primary competitive arena. IQGeo's Comsof Fiber is the closest competitor, offering automated fiber network planning. However, Comsof uses heuristic-based optimization, not machine learning. It cannot learn from historical data, predict failures, or adapt to changing conditions. AtlasPro AI's GNN-based approach enables predictive analytics that are fundamentally impossible with rule-based systems.

3-GIS provides comprehensive network management but lacks AI capabilities. Their strength is in data management and visualization, not intelligent analysis. AtlasPro AI can integrate with 3-GIS data while providing the AI layer that 3-GIS lacks.

5.3 Category 3: AI-Powered Geospatial Intelligence

Analysis. World Labs, founded by Fei-Fei Li, is building frontier spatial AI models. However, their focus is on 3D world generation for consumer and creative applications, not B2B infrastructure. They represent a future threat if they pivot to enterprise, but their current trajectory is divergent from AtlasPro AI's focus.

Orbital Insight and Planet Labs focus on satellite imagery analysis for geopolitical and envi-

Table 4: AI-Powered Geospatial Intelligence Competitors

Company	Funding	Overlap	AtlasPro Differentiation
World Labs	$230M	Future (5/10)	B2B infrastructure vs. consumer 3D generation
Orbital Insight	$128M	Adjacent (3/10)	Network graph focus vs. satellite imagery
Planet Labs	Public	Adjacent (2/10)	Infrastructure intelligence vs. Earth observation
Blackshark.ai	$20M	Adjacent (3/10)	Network topology vs. 3D city models

ronmental intelligence. They lack expertise in network topology and infrastructure-specific applications. AtlasPro AI's GNN-based approach is specifically designed for graph-structured network data, not raster imagery.

5.4 Category 4: Graph Analytics Platforms

Table 5: Graph Analytics Platform Competitors

Company	Funding	Overlap	AtlasPro Differentiation
Neo4j	$325M	Partial (6/10)	Vertical solution vs. horizontal graph database
TigerGraph	$105M	Partial (5/10)	Domain expertise vs. general graph analytics
Amazon Neptune	N/A	Partial (4/10)	Specialized models vs. generic graph storage

Analysis. Graph database platforms like Neo4j provide the infrastructure for storing and querying graph data. They have added spatial capabilities (Neo4j Spatial) and some machine learning features. However, they are horizontal platforms without domain-specific expertise. AtlasPro AI can use Neo4j or similar platforms as a data layer while providing the specialized GNN models and agentic workflows that these platforms lack.

5.5 Category 5: Spatial Data Platforms

Analysis. Wherobots, built on Apache Sedona, provides serverless spatial data processing. They are a potential integration partner rather than a direct competitor. AtlasPro AI's value proposition is the AI reasoning layer, not the data processing infrastructure. We can build on top of Wherobots or similar platforms.

6 AtlasPro AI's Unique Position

Our competitive analysis confirms that AtlasPro AI is creating a new category at the intersection of agentic AI and network infrastructure intelligence. Our differentiation is not a single feature,

Table 6: Spatial Data Platform Competitors

Company	Funding	Overlap	AtlasPro Differentiation
Wherobots	$15M	Partial (5/10)	Agentic reasoning vs. spatial data processing
Snowflake	Public	Indirect (3/10)	Specialized AI vs. general data warehouse
Databricks	$4.1B	Indirect (3/10)	Vertical focus vs. horizontal ML platform

but a combination of architectural and business model choices that are difficult for competitors to replicate.

6.1 The Four Pillars of Differentiation

Pillar 1: GNN + MCP Integration. This is our core technical differentiation. We are the only company combining graph neural networks for network topology reasoning with the Model Context Protocol for AI agent tool use. Incumbent GIS and network planning tools lack the architectural foundation for GNNs. AI platform companies lack the domain-specific data and models for network topology. AtlasPro AI is unique in combining both.

Pillar 2: Vertical Focus. Our deep focus on telecom and utilities allows us to build highly specialized models and workflows that solve high-value problems for a specific customer profile. This is a classic "vertical SaaS" advantage over horizontal platforms. We understand the language, workflows, and pain points of network engineers in a way that horizontal AI platforms cannot.

Pillar 3: AI-Native Architecture. AtlasPro AI is built from the ground up for agentic workflows. This is fundamentally different from retrofitting AI onto a 20-year-old GIS architecture. Our system is designed for agents to query, plan, and act on network data, a paradigm shift from human-in-the-loop analysis.

Pillar 4: Data Network Effects. As we deploy with customers like Google Fiber and Ziply Fiber, we gain access to unique network data that allows us to train increasingly sophisticated GNN models. This creates a powerful data advantage over time, as our models become more accurate and predictive than any competitor's.

AtlasPro AI Strategic Differentiation

Our defensibility comes from the unique combination of a vertically-focused GNN-powered platform with an AI-native, agentic architecture. This allows us to solve high-value problems in critical infrastructure sectors that are inaccessible to both horizontal AI platforms and legacy GIS incumbents.

Table 7: Competitive Differentiation Comparison

Capability	Esri	IQGeo	Neo4j	World Labs	AtlasPro
GNN-based reasoning	No	No	Partial	No	Yes
Agentic workflows	No	No	No	Partial	Yes
MCP integration	No	No	No	No	Yes
Telecom expertise	Partial	Yes	No	No	Yes
Predictive analytics	Partial	No	Partial	No	Yes

6.2 Competitive Differentiation Analysis

Part III
Technical Foundations: Agentic AI for Spatial Intelligence

7 Agentic Architectures

7.1 The ReAct Paradigm

The ReAct framework [Yao et al., 2023] established the foundation for modern LLM agents by interleaving reasoning and acting:

1. **Thought:** The agent reasons about the current state and what action to take.
2. **Action:** The agent executes an action (e.g., tool call, API request).
3. **Observation:** The agent receives feedback from the environment.
4. **Repeat:** The cycle continues until the task is complete.

For spatial tasks, ReAct-style agents must ground their reasoning in geometric reality. A navigation agent might think: "I need to reach the kitchen. Based on the map, I should turn left at the hallway." The action is a movement command, and the observation is the new visual input.

7.2 Reflexion and Self-Improvement

Reflexion [Shinn et al., 2023] extends ReAct with self-reflection:

1. Execute task using ReAct
2. Evaluate outcome (success/failure)
3. Generate reflection on what went wrong

4. Store reflection in memory
5. Retry task with reflection as additional context

For spatial agents, reflection is critical for learning from geometric mistakes. An agent that collides with an obstacle can reflect: "I underestimated the width of the doorway. Next time, I should add a safety margin."

7.3 Multi-Agent Coordination

Complex spatial tasks often require multiple agents working together. Key frameworks include:

AutoGen [Wu et al., 2023]: Enables multi-agent conversations where agents can delegate tasks to each other.

MetaGPT [Hong et al., 2023]: Assigns roles to agents (e.g., planner, executor, critic) for structured collaboration.

CAMEL [Li et al., 2023]: Uses role-playing to enable emergent collaboration between agents.

For AtlasPro AI, multi-agent coordination is relevant for complex network planning tasks where different agents might handle different aspects (e.g., capacity planning, routing optimization, failure prediction).

8 Memory Systems for Spatial Agents

Memory is the foundation of intelligent behavior. For spatial agents, we identify three memory tiers with distinct computational and representational requirements.

8.1 Short-Term Memory: Context Management

Short-term memory operates within the LLM's context window. Modern LLMs have context windows ranging from 8K to 128K+ tokens [OpenAI, 2023, Anthropic, 2024]. For spatial tasks, we must efficiently encode:

- Current observations (images, sensor data)
- Recent action history
- Task instructions and goals
- Relevant retrieved information

State Compression. For long-horizon tasks, we must compress historical state to fit within context limits. Techniques include summarization of past events, selective retention of important information, and hierarchical state representations.

8.2 Long-Term Memory: Retrieval-Augmented Generation

Long-term memory extends agent knowledge beyond the context window through external retrieval [Lewis et al., 2020, Guu et al., 2020].

Vector Database Selection. Key options for production deployment include:

- **Pinecone:** Managed service, easy scaling, good for production

- **Weaviate:** Open-source, supports hybrid search
- **Chroma:** Lightweight, good for prototyping
- **Milvus:** High-performance, supports billion-scale vectors

Embedding Model Selection. The choice of embedding model affects retrieval quality. Options include OpenAI text-embedding-3-large for general performance, Sentence-BERT variants for semantic similarity, and domain-specific embeddings for specialized tasks.

8.3 Spatial Memory: Cognitive Maps

Spatial memory requires specialized representations that encode geometric relationships. Key approaches include:

VLMaps [Huang et al., 2023a]: Creates language-indexed spatial maps by projecting CLIP features into a 3D voxel grid. Enables natural language queries like "Where is the refrigerator?"

Neural SLAM [Chaplot et al., 2020]: Learns to build spatial maps and localize within them end-to-end.

Scene Graphs [Armeni et al., 2019]: Represents spatial relationships as graphs with objects as nodes and relationships as edges.

For AtlasPro AI, spatial memory takes the form of network topology graphs with rich node and edge attributes.

9 Planning Under Geometric Constraints

9.1 The Planning Challenge

LLMs can generate high-level plans but struggle with geometric constraints [Valmeekam et al., 2023, Kambhampati et al., 2024]. Consider a robot tasked with "Put the cup in the cabinet." An LLM might generate:

1. Pick up the cup
2. Open the cabinet
3. Place the cup inside
4. Close the cabinet

This plan is semantically correct but geometrically naive. It doesn't account for:

- The robot's current position relative to the cup and cabinet
- Whether the cabinet is reachable from the current position
- The grasp pose required for the cup
- Collision avoidance during movement

9.2 Task and Motion Planning (TAMP)

TAMP systems [Garrett et al., 2021] combine symbolic planning with geometric reasoning:

Symbolic Layer. Plans at the level of actions and predicates (e.g., "holding(cup)", "open(cabinet)").

Geometric Layer. Computes feasible configurations, trajectories, and grasps.

Integration. The symbolic planner proposes actions; the geometric planner verifies feasibility and computes parameters.

9.3 LLM-Based Planning

Recent work explores using LLMs directly for planning:

SayCan [Ahn et al., 2022]: Grounds LLM plans in robot affordances by scoring actions based on both semantic relevance (from LLM) and feasibility (from learned value functions).

Code as Policies [Liang et al., 2023]: Generates executable Python code that calls robot APIs, enabling complex behaviors through code composition.

LLM+P [Liu et al., 2023a]: Uses LLMs to generate PDDL problem specifications, then solves with classical planners.

9.4 World Model-Based Planning

World models enable planning through imagination [Hafner et al., 2023, Ha and Schmidhuber, 2018]:

1. Learn a model of environment dynamics
2. Simulate future states given actions
3. Optimize actions to achieve goals in imagination
4. Execute best action sequence in real world

This approach is particularly valuable for safety-critical applications where trial-and-error in the real world is unacceptable.

10 Tool Use and Action

10.1 The Tool Use Paradigm

Tool use extends agent capabilities beyond the LLM's intrinsic abilities [Schick et al., 2023, Patil et al., 2023]. For spatial agents, relevant tools include:

- **Perception APIs:** Object detection, depth estimation, semantic segmentation
- **Navigation APIs:** Path planning, localization, mapping
- **Manipulation APIs:** Grasp planning, motion planning, force control
- **GIS APIs:** Spatial queries, routing, geocoding

10.2 Model Context Protocol (MCP)

The Model Context Protocol provides a standardized interface for LLM agents to interact with external tools. For AtlasPro AI, MCP is the bridge between our GNN-powered backend and LLM-based reasoning.

MCP Architecture:

1. **Tool Registry:** Defines available tools with schemas
2. **Tool Invocation:** Agent calls tools with structured parameters
3. **Result Handling:** Tool results are returned to agent context

We detail AtlasPro AI's MCP integration strategy in Part V.

Part IV
Enabling Technologies

11 Graph Neural Networks for Spatial Reasoning

Graph Neural Networks are the cornerstone of AtlasPro AI's technical approach. Traditional machine learning models struggle with the non-Euclidean, relational structure of network infrastructure. GNNs are uniquely suited to this domain.

11.1 Why GNNs for Network Infrastructure?

Telecommunication and utility networks are fundamentally graphs. Nodes represent connection points (e.g., splice closures, utility poles, substations) and edges represent physical connections (e.g., fiber optic cables, power lines). While traditional machine learning models like Multi-Layer Perceptrons (MLPs) or even powerful sequence models like Transformers can be applied to this data, they fail to capture the inherent topological structure, leading to suboptimal performance. GNNs, by contrast, are designed with inductive biases that are perfectly suited to this domain.

11.1.1 Theoretical Justification: Inductive Biases for Spatial Data

The superiority of GNNs for network data stems from their inherent inductive biases, which align with the fundamental properties of spatial networks:

1. Permutation Equivariance. The output of a GNN is equivariant to the ordering of nodes in the adjacency matrix. This means that if we re-order the nodes, the output node representations are re-ordered in the same way. This is a critical property for network data, where the node ordering is arbitrary. An MLP, by contrast, would produce a completely different output if the node ordering is changed, making it unsuitable for this type of data.

2. Locality. GNNs operate on local neighborhoods, aggregating information from a node's immediate neighbors. This aligns with the principle of spatial locality, where nearby entities are more likely to influence each other. For example, a fault in a fiber optic cable is most likely to affect the adjacent connection points. Transformers, while powerful, have a global attention mechanism that can be computationally expensive and may not be necessary for many spatial tasks.

3. Compositionality. GNNs can learn compositional representations of complex structures. For example, a GNN can learn to represent a "ring" topology in a fiber network by composing the

representations of individual nodes and edges. This allows GNNs to generalize to unseen network configurations.

11.1.2 Empirical Evidence from Literature

Published research provides strong empirical evidence for the superiority of GNNs on graph-structured data:

- A recent study in *Nature Communications* [**?**] demonstrated that GNNs achieve 40% higher accuracy than MLPs in predicting network layout properties.
- In the context of traffic forecasting, a spatio-temporal GNN (DCRNN) achieved a 15-20% reduction in prediction error compared to traditional time-series models [Li et al., 2018].
- For node classification tasks on benchmark graph datasets like Cora and Citeseer, GCNs and GATs consistently outperform MLPs and other non-graph-based methods by a significant margin [Kipf and Welling, 2017, Velickovic et al., 2018].

11.1.3 Why Not Other Architectures?

Multi-Layer Perceptrons (MLPs) treat each node independently, ignoring the rich topological information in the network. This is like trying to understand a city by looking at a list of buildings without a map.

Convolutional Neural Networks (CNNs) are designed for grid-like data (e.g., images) and cannot be directly applied to the irregular, non-Euclidean structure of network graphs.

Transformers have a global attention mechanism that can be computationally expensive for large graphs. While some graph-based Transformers have been proposed, they often require clever positional encodings to incorporate structural information, which is a natural byproduct of GNNs.

11.1.4 Application to AtlasPro AI's Use Case

For AtlasPro AI's focus on dense, unstructured location data in telecommunications and utilities, GNNs are the ideal choice because:

- They can naturally handle the complex, non-grid-like topology of fiber optic and power networks.
- They can integrate heterogeneous data sources (e.g., cable type, pole age, maintenance history) as node and edge features.
- They can learn to predict system-level properties (e.g., network-wide failure risk) from local node and edge information.
- They can scale to networks with millions of nodes and edges through techniques like neighborhood sampling (GraphSAGE).

In summary, the choice of GNNs as the core of AtlasPro AI's technical approach is not arbitrary; it is a principled decision based on the fundamental alignment between the inductive biases of GNNs and the properties of spatial network data. This provides a significant and sustainable technical advantage over alternative approaches.

Telecommunication and utility networks are fundamentally graphs. Nodes represent connection points (e.g., splice closures, utility poles, substations) and edges represent physical connections (e.g., fiber optic cables, power lines). GNNs allow us to:

- **Model Topology:** Directly learn from the network's structure
- **Incorporate Features:** Combine topological information with rich data like cable capacity, pole age, or maintenance history
- **Predictive Analytics:** Forecast network failures, predict capacity bottlenecks, and optimize expansion plans
- **Scalability:** Handle networks with millions of nodes and edges

11.2 Core GNN Architectures

11.2.1 Graph Convolutional Networks (GCN)

GCN [Kipf and Welling, 2017] introduced spectral graph convolutions:

$$H^{(l+1)} = \sigma\left(\tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}H^{(l)}W^{(l)}\right) \tag{1}$$

where $\tilde{A} = A + I$ is the adjacency matrix with self-loops, $\tilde{D}$ is the degree matrix, $H^{(l)}$ is the feature matrix at layer l, and $W^{(l)}$ is a learnable weight matrix.

Spatial Interpretation. Each node aggregates features from its neighbors, weighted by degree. This is analogous to spatial smoothing: nodes become similar to their neighbors.

11.2.2 Graph Attention Networks (GAT)

GAT [Velickovic et al., 2018] introduces attention mechanisms:

$$h_i^{(l+1)} = \sigma\left(\sum_{j\in\mathcal{N}(i)} \alpha_{ij} W h_j^{(l)}\right) \tag{2}$$

where α_{ij} are learned attention coefficients.

Advantages for Network Data. GAT can learn which connections are most important. For a fiber network, this might mean prioritizing backbone connections over local drops.

11.2.3 GraphSAGE

GraphSAGE [Hamilton et al., 2017] enables inductive learning through neighborhood sampling:

1. Sample fixed-size neighborhood
2. Aggregate neighbor features
3. Concatenate with node's own features
4. Apply neural network

Scalability. GraphSAGE scales to large graphs through mini-batch training, essential for production network data.

11.3 Spatio-Temporal GNNs

For time-varying spatial data (traffic, network load, failure patterns):

DCRNN [Li et al., 2018]: Models traffic as diffusion on road graph with GRU for temporal modeling.

STGCN [Yu et al., 2018]: Separates spatial and temporal convolutions for efficiency.

Graph WaveNet [Wu et al., 2019]: Learns adaptive adjacency matrix with dilated causal convolutions.

11.4 GNN-LLM Integration Patterns

11.4.1 Pattern 1: GNN as Encoder

Use GNN to encode graph structure, pass to LLM:

1. GNN encodes graph → node embeddings
2. Project embeddings to LLM token space
3. Concatenate with text tokens
4. LLM processes combined input

Example: GraphGPT [Tang et al., 2024]: Graph encoder aligned with LLM for graph-based question answering.

11.4.2 Pattern 2: LLM for Graph Enhancement

Use LLM to improve GNN:

- Generate node features from text descriptions
- Explain GNN predictions
- Augment training data

11.4.3 Pattern 3: GNN-RAG

Use GNN for knowledge graph retrieval [Wang et al., 2024a]:

1. Query → retrieve relevant subgraph
2. GNN reasons over subgraph
3. Linearize subgraph for LLM
4. LLM generates final answer

This pattern is particularly relevant for AtlasPro AI, where we retrieve relevant network subgraphs to answer user queries.

12 World Models for Safe Deployment

World models learn predictive models of environment dynamics, enabling planning through imagination rather than trial-and-error [Ha and Schmidhuber, 2018].

12.1 The Dreamer Series

Dreamer [Hafner et al., 2019]: Learns latent dynamics model, plans in imagination, actor-critic in latent space.

DreamerV2 [Hafner et al., 2021]: Discrete latent representations, human-level Atari performance, more stable training.

DreamerV3 [Hafner et al., 2023]: Single algorithm across domains, symlog predictions for stability, fixed hyperparameters.

Conceptual Progression. The Dreamer series demonstrates that world models can achieve strong performance across diverse domains with a single architecture. This is significant for AtlasPro AI: we can potentially use a single world model architecture across different network types (fiber, power, water).

12.2 Video World Models

Genie [Bruce et al., 2024]: Learns controllable world models from internet videos, generates interactive environments, enables training without simulators.

GAIA-1 [Hu et al., 2023]: World model for autonomous driving, generates realistic driving videos conditioned on actions.

Sora [Brooks et al., 2024]: OpenAI's video generation model demonstrates emergent world simulation capabilities.

12.3 LLM-Based World Models

RAP [Hao et al., 2023]: Uses LLM as world model with Monte Carlo Tree Search for planning.

Limitations. LLMs may hallucinate state transitions, need grounding in real observations, and uncertainty quantification is challenging.

13 Vision-Language-Action Models

VLAs represent the convergence of vision, language, and action in a single model [Brohan et al., 2023].

13.1 RT-2: Robotic Transformer 2

RT-2 [Brohan et al., 2023] treats robot actions as text tokens:

- Vision-language model backbone (PaLI-X or PaLM-E)
- Actions tokenized as text (e.g., "[0.1, 0.2, -0.05, ...]")
- End-to-end training on robot data
- Emergent capabilities from VLM pretraining

13.2 Open-Source VLAs

Octo [Team et al., 2024]: Generalist robot policy trained on Open X-Embodiment dataset, supports multiple robots and tasks.

OpenVLA [Kim et al., 2024]: Open-source VLA with strong performance, available weights and training code.

π_0 [Black et al., 2024]: Physical Intelligence's foundation model for robotics.

13.3 Relevance to AtlasPro AI

While VLAs are designed for physical robot control, the architecture pattern is relevant:

- Multimodal input (network diagrams, satellite imagery, sensor data)
- Language understanding (natural language queries)
- Action output (network configuration changes, maintenance recommendations)

14 Geospatial Foundation Models

14.1 Remote Sensing Models

Prithvi [Jakubik et al., 2024]: NASA/IBM geospatial foundation model pretrained on Harmonized Landsat Sentinel-2 data. Supports land use classification, flood mapping, wildfire detection, and crop monitoring.

SatMAE [Cong et al., 2022]: Self-supervised learning for satellite imagery using masked autoencoder approach.

SatlasPretrain [Bastani et al., 2023]: Large-scale pretraining on 302M image dataset with multiple sensor types.

14.2 Urban Computing

Traffic Prediction. leading approaches use graph-based spatial modeling with temporal sequence modeling [Jin et al., 2023, Li et al., 2018, Yu et al., 2018].

Smart City Applications. Traffic management, energy optimization, public safety, and urban planning [Zheng et al., 2014].

Part V
AtlasPro AI's Technical Approach

15 Architectural Principles

Based on our analysis of over 800 papers and the competitive landscape, we establish six architectural principles to guide our system development.

Principle 1: Explicit Spatial Representation. Systems must maintain an explicit, geometrically-grounded representation of the network, separate from linguistic representations. For AtlasPro AI, this means maintaining a graph database with precise geographic coordinates, connectivity information, and physical attributes.

Principle 2: Hybrid Symbolic-Neural Planning. Planning must combine the flexibility of neural models with the rigor of symbolic methods for constraint satisfaction. Network planning involves hard constraints (e.g., cable capacity limits, regulatory requirements) that must be respected.

Principle 3: Graph-Based Reasoning. Graph neural networks are a core component for reasoning about spatial relationships and network structures. This is our primary technical differentiator.

Principle 4: Hierarchical Memory. Memory systems must operate at multiple time scales, from short-term context to long-term episodic memory and persistent spatial knowledge. Network data accumulates over years; our system must leverage this history.

Principle 5: Safety through World Models. Predictive world models are essential for safe planning and decision-making. Before recommending a network change, we should simulate its effects.

Principle 6: Continuous Evaluation. Internal benchmarking and red teaming are integral to the development process, not an afterthought.

16 GNN Architecture for Network Intelligence

16.1 AtlasPro's GNN Architecture (Conceptual)

Our architecture is a spatio-temporal GNN designed for infrastructure networks.

Spatial Component. We use a Graph Attention Network (GAT) to weigh the importance of different connections. For example, a high-capacity backbone fiber connection is more important for network-wide analysis than a local drop cable.

Temporal Component. We use a Gated Recurrent Unit (GRU) or Transformer-based architecture to model how the network changes over time. This allows us to predict future states based on historical maintenance data, weather patterns, and demand growth.

Multi-Scale Aggregation. We use hierarchical pooling to aggregate information from local neighborhoods to regional clusters to the entire network.

16.2 Implementation Approach

```python
import torch
from torch_geometric.nn import GATConv, global_mean_pool

class AtlasProGNN(torch.nn.Module):
    def __init__(self, in_channels, hidden_channels, out_channels):
        super().__init__()
        # Spatial encoding with attention
        self.gat1 = GATConv(in_channels, hidden_channels, heads=4)
        self.gat2 = GATConv(hidden_channels * 4, hidden_channels, heads=4)

        # Temporal encoding
        self.temporal = torch.nn.GRU(hidden_channels * 4, hidden_channels,
    batch_first=True)

        # Output head
        self.classifier = torch.nn.Linear(hidden_channels, out_channels)

    def forward(self, x, edge_index, batch, temporal_features=None):
        # Spatial message passing
        x = self.gat1(x, edge_index).relu()
        x = self.gat2(x, edge_index)

        # Temporal processing (if temporal features provided)
        if temporal_features is not None:
```

```
            x, _ = self.temporal(temporal_features)
            x = x[:, -1, :]  # Take last timestep

        # Global pooling and classification
        x = global_mean_pool(x, batch)
        return self.classifier(x)
```

Listing 1: Conceptual AtlasPro GNN Architecture

17 MCP Integration for Agentic Spatial Reasoning

The Model Context Protocol (MCP) is the interface that allows our AI agents to interact with the GNN-powered backend. It exposes the capabilities of our spatial intelligence platform as a set of tools that an LLM agent can use.

17.1 The Role of MCP

MCP standardizes how an LLM communicates with external tools. For AtlasPro AI, this means our GNN models and spatial databases are wrapped in an MCP-compliant API. An AI agent can then perform complex spatial analysis by calling these tools.

17.2 Example MCP Tool Definitions

Tool: find_capacity_bottlenecks

```
{
  "name": "find_capacity_bottlenecks",
  "description": "Identifies fiber segments in a given area that are nearing
    capacity.",
  "parameters": {
    "area_polygon": {
      "type": "geojson",
      "description": "Polygon defining the area to analyze."
    },
    "capacity_threshold": {
      "type": "number",
      "description": "Capacity utilization percentage (e.g., 80)."
    }
  }
}
```

Listing 2: MCP Tool Definition: Capacity Bottlenecks

Tool: predict_failure_risk

```
{
  "name": "predict_failure_risk",
  "description": "Predicts the failure risk for network components using GNN model
    .",
  "parameters": {
    "component_type": {
      "type": "string",
      "enum": ["cable", "pole", "splice", "substation"]
    },
    "time_horizon_days": {
      "type": "integer",
      "description": "Number of days to forecast."
```

```
    }
  }
}
```

Listing 3: MCP Tool Definition: Failure Risk Prediction

Tool: optimize_routing

```
{
  "name": "optimize_routing",
  "description": "Generates optimal routing for new fiber connections.",
  "parameters": {
    "source_location": {
      "type": "geojson",
      "description": "Starting point for the connection."
    },
    "destination_location": {
      "type": "geojson",
      "description": "Ending point for the connection."
    },
    "constraints": {
      "type": "object",
      "description": "Routing constraints (avoid areas, max distance, etc.)"
    }
  }
}
```

Listing 4: MCP Tool Definition: Routing Optimization

17.3 Agentic Workflow Example

An LLM agent can chain these tools together to perform complex tasks.

User Prompt: "Show me the highest-risk fiber cables in downtown that are over 80% capacity and suggest a rerouting plan."

Agent's Internal Reasoning (ReAct-style):

1. **Thought:** I need to find high-capacity, high-risk cables in the "downtown" area. First, I need the polygon for downtown.
2. **Action:** `get_area_polygon(name="downtown")`
3. **Observation:** (Returns GeoJSON polygon)
4. **Thought:** Now I can find the capacity bottlenecks.
5. **Action:** `find_capacity_bottlenecks(area_polygon=..., capacity_threshold=80)`
6. **Observation:** (Returns list of 12 high-capacity cables)
7. **Thought:** Now I need to find the failure risk for these specific cables.
8. **Action:** `predict_failure_risk(component_ids=[...], time_horizon_days=90)`
9. **Observation:** (Returns list of cables with risk scores; 3 cables have risk > 0.7)
10. **Thought:** I have identified 3 high-risk, high-capacity cables. Now I need to generate rerouting plans.

11. **Action:** `optimize_routing(cables_to_reroute=[...])`
12. **Observation:** (Returns proposed new fiber paths with cost estimates)
13. **Thought:** I have all the information. I will now synthesize the answer for the user.

This agentic workflow, enabled by our GNN backend and MCP tool interface, is a core differentiator that legacy systems cannot replicate.

18 Failure Mode Analysis

Our research identified six systematic failure modes that plague existing spatial AI systems. AtlasPro AI's architecture is designed to mitigate each.

18.1 Failure Mode 1: Spatial Hallucination

Definition. The system generates spatially impossible outputs, such as paths through obstacles or connections that violate physical constraints.

Cause. LLMs lack grounded spatial representations and may generate plausible-sounding but geometrically invalid outputs.

AtlasPro Mitigation. Our GNN-based backend maintains an explicit graph representation of the network. All outputs are validated against this ground truth before being returned to the user.

18.2 Failure Mode 2: Scale Confusion

Definition. The system applies methods appropriate for one scale to a different scale, leading to errors.

Cause. Methods optimized for micro-scale manipulation don't transfer to macro-scale network planning.

AtlasPro Mitigation. Our three-axis taxonomy explicitly encodes scale. Our system uses different models and representations for different scales, with explicit handoffs between them.

18.3 Failure Mode 3: Temporal Incoherence

Definition. The system fails to maintain consistent state over time, leading to contradictory recommendations.

Cause. LLM context windows are limited; long-horizon tasks exceed memory capacity.

AtlasPro Mitigation. Our hierarchical memory architecture combines short-term context with long-term retrieval and persistent spatial knowledge in the graph database.

18.4 Failure Mode 4: Constraint Violation

Definition. The system generates outputs that violate hard constraints (e.g., capacity limits, regulatory requirements).

Cause. Neural models are soft optimizers; they don't naturally respect hard constraints.

AtlasPro Mitigation. Our hybrid symbolic-neural planning approach uses neural models for optimization but validates all outputs against symbolic constraint checkers.

18.5 Failure Mode 5: Compositional Failure

Definition. The system succeeds on simple tasks but fails on complex tasks that require composing multiple capabilities.

Cause. Training data may not cover all possible task compositions.

AtlasPro Mitigation. Our agentic architecture decomposes complex tasks into simpler subtasks, each handled by specialized tools. The LLM orchestrates the composition.

18.6 Failure Mode 6: Distribution Shift Fragility

Definition. The system performs well on training distribution but fails on novel inputs.

Cause. Neural models don't generalize well to out-of-distribution inputs.

AtlasPro Mitigation. Our data flywheel continuously incorporates new data from customer deployments. We also implement uncertainty quantification to flag low-confidence predictions.

Part VI
Research Roadmap and Conclusion

19 Three-Phase Research Roadmap

19.1 Phase 1: Foundation (Q1-Q2 2026)

Objective: Establish core infrastructure and validate technical approach.

Deliverables:

- Graph database infrastructure with customer data integration
- Baseline GNN models for network representation learning
- MCP tool definitions and initial agent integration
- Internal benchmarking framework

Success Metrics:

- GNN model achieves >80% accuracy on failure prediction task
- Agent successfully completes 10 representative customer queries
- Benchmark suite covers 5 core use cases

19.2 Phase 2: Capability Expansion (Q3-Q4 2026)

Objective: Expand capabilities and begin customer pilots.

Deliverables:

- Spatio-temporal GNN for predictive analytics
- World model for network simulation
- Multi-agent coordination for complex planning tasks

- Customer pilot deployments (3-5 customers)

Success Metrics:

- Predictive model achieves >70% precision on 90-day failure forecast
- World model accurately simulates network changes
- Pilot customers report >30% efficiency improvement

19.3 Phase 3: Scale and Productization (2027)

Objective: Scale to production and establish market leadership.

Deliverables:

- Production-grade platform with SLA guarantees
- Self-improving system with continuous learning
- Expanded vertical coverage (utilities, smart cities)
- Published research establishing thought leadership

Success Metrics:

- 20+ production customers
- $5M+ ARR
- 2+ publications at top-tier venues
- Industry recognition as category leader

20 Grand Challenges for the Field

We identify six grand challenges that represent fundamental bottlenecks for spatial AI:

Challenge 1: Unified Spatial Representation. How can agents maintain a single, coherent spatial representation that supports reasoning across micro, meso, and macro scales?

Challenge 2: Grounded Long-Horizon Planning. How can agents plan over extended horizons while maintaining geometric feasibility?

Challenge 3: Safe Deployment Under Uncertainty. How can spatial AI systems operate safely in safety-critical applications with guaranteed bounds on failure?

Challenge 4: Sim-to-Real Transfer. How can policies learned in simulation transfer reliably to the physical world?

Challenge 5: Scalable Multi-Agent Coordination. How can large numbers of spatial agents coordinate effectively with limited communication?

Challenge 6: Efficient Edge Deployment. How can capable spatial AI systems run on resource-constrained platforms?

21 Limitations of This Report

This report has several limitations that we acknowledge:

- Our paper selection process, though systematic, may have missed relevant works in adjacent fields or non-English publications.
- The proposed taxonomy is one of many possible categorizations and may not capture all nuances of the field.
- Our competitive analysis is based on publicly available information and may not reflect the latest developments at private companies.
- The architectural principles and roadmap are preliminary and will evolve as we gain more data and customer feedback.
- We have not yet validated our approach with production deployments; the claims in this report are based on our analysis of the literature and preliminary experiments.

22 Conclusion

This technical report has presented AtlasPro AI's comprehensive research approach to building autonomous spatial intelligence systems for critical infrastructure. Our key contributions include:

1. A **unified three-axis taxonomy** (Task × Capability × Scale) that organizes the design space for spatial AI agents.
2. A **comprehensive competitive analysis** demonstrating AtlasPro AI's unique market position at the intersection of agentic AI and network intelligence.
3. A **systematic analysis** of failure modes that inform our architectural decisions.
4. **Architectural principles** and a **research roadmap** for building production-grade spatial AI systems.

We believe that spatial intelligence represents the next frontier for AI systems. The ability to perceive, reason about, and act within physical environments is essential for AI to move beyond the digital realm and have meaningful impact in the real world. AtlasPro AI is committed to advancing this frontier, with an initial focus on the critical infrastructure sectors where our team has deep expertise and where the need for intelligent automation is most acute.

We release this report to establish priority on our methodological contributions and to invite collaboration from the research community. We welcome feedback, partnerships, and opportunities to advance the field together.

Acknowledgments. We thank our advisors, investors, and early customers for their support and feedback. We also thank the broader research community whose work forms the foundation of this report.

Appendices

1 Comprehensive Benchmark Analysis

This appendix provides detailed analysis of key benchmarks for spatial AI evaluation, organized by task domain.

1.1 Navigation Benchmarks

1.1.1 Room-to-Room (R2R)

The Room-to-Room dataset [Anderson et al., 2018] is the foundational benchmark for vision-language navigation:

- **Environment:** Matterport3D scans of 90 real buildings
- **Task:** Follow natural language instructions to navigate to goal locations
- **Statistics:** 7,189 paths, average path length 10m, 6 viewpoints per path
- **Metrics:** Success Rate (SR), SPL, Navigation Error (NE)

Table 8: R2R Val-Unseen Leaderboard (Top Methods)

Method	SR (%)	SPL (%)	NE (m)
Human Performance	86	76	1.61
DUST [Chen et al., 2024b]	72	62	3.12
HOPT [Qiao et al., 2023]	64	57	3.89
Recurrent VLN-BERT [Hong et al., 2021]	63	57	3.93
VLN-BERT [Majumdar et al., 2020]	61	55	4.09

Analysis. Despite significant progress, there remains a substantial gap between human performance (86% SR) and the best models (72% SR). This gap is particularly pronounced in unseen environments, indicating that current models struggle with generalization.

1.1.2 Room-across-Room (RxR)

RxR [Ku et al., 2020] extends R2R with multilingual instructions and longer paths:

- **Languages:** English, Hindi, Telugu
- **Statistics:** 126,069 instruction-path pairs
- **Key Difference:** Instructions are more detailed and paths are longer

1.1.3 REVERIE

REVERIE [Qi et al., 2020] adds object grounding to navigation:

- **Task:** Navigate to a room and identify a specific object
- **Challenge:** Requires both navigation and object recognition
- **Statistics:** 21,702 instructions across 4,140 target objects

1.1.4 Habitat Challenge

The annual Habitat Challenge [Savva et al., 2019, Szot et al., 2021, Puig et al., 2024] provides standardized evaluation:

- **ObjectNav:** Navigate to instances of object categories
- **PointNav:** Navigate to specified coordinates
- **Social Navigation:** Navigate among humans (Habitat 3.0)

1.2 Manipulation Benchmarks

1.2.1 RLBench

RLBench [James et al., 2020] provides 100 unique manipulation tasks:

- **Simulation:** CoppeliaSim with Franka Panda robot
- **Tasks:** Reach, push, pick-and-place, tool use, assembly
- **Variations:** Multiple variations per task (colors, positions, objects)
- **Metrics:** Task success rate, episode length

Table 9: RLBench Multi-Task Performance (10 Tasks, 100 Demos Each)

Method	Avg. Success (%)	Training Time
RVT-2 [Goyal et al., 2024]	81.4	8 hours
RVT [Goyal et al., 2023]	62.9	12 hours
PerAct [Shridhar et al., 2023]	49.4	16 hours
C2F-ARM [James et al., 2022]	39.2	24 hours

1.2.2 Meta-World

Meta-World [Yu et al., 2020] focuses on multi-task and meta-learning:

- **Robot:** Sawyer arm simulation
- **Tasks:** 50 distinct manipulation tasks
- **Benchmarks:** ML1 (single-task), ML10 (10 tasks), ML45 (45 tasks), MT50 (multi-task)

1.2.3 CALVIN

CALVIN [Mees et al., 2022] evaluates language-conditioned manipulation:

- **Task:** Execute sequences of language instructions
- **Challenge:** Long-horizon, compositional tasks
- **Metric:** Average chain length (how many consecutive instructions completed)

1.3 Agent Benchmarks

1.3.1 AgentBench

AgentBench [Liu et al., 2023c] provides comprehensive LLM agent evaluation:

- **Environments:** 8 distinct environments
- **Categories:** Operating system, database, knowledge graph, web browsing, lateral thinking, card game, digital card game, house-holding
- **Metrics:** Task success rate, efficiency

Table 10: AgentBench Overall Performance

Model	Overall Score	Best Environment
GPT-4	4.01	Web Browsing
Claude-2	3.12	Operating System
GPT-3.5-Turbo	2.89	Database
LLaMA-2-70B	1.54	Knowledge Graph

1.3.2 EmbodiedBench

EmbodiedBench [Yang et al., 2025] evaluates embodied multimodal LLMs:

- **Tasks:** Navigation, manipulation, spatial reasoning
- **Evaluation:** Both perception and action capabilities
- **Key Finding:** Current MLLMs struggle with spatial reasoning

1.3.3 SafeAgentBench

SafeAgentBench [Yin et al., 2025] focuses on safety evaluation:

- **Focus:** Safety-critical scenarios
- **Metrics:** Safety rate, task completion under constraints
- **Scenarios:** Collision avoidance, constraint satisfaction

1.4 Autonomous Driving Benchmarks

1.4.1 nuScenes

nuScenes [Caesar et al., 2020] is the standard benchmark for 3D perception:

- **Data:** 1000 scenes, 1.4M camera images, 390K lidar sweeps
- **Sensors:** 6 cameras, 1 lidar, 5 radars, GPS/IMU
- **Annotations:** 1.4M 3D bounding boxes, 23 object classes
- **Metrics:** mAP, NDS (nuScenes Detection Score)

1.4.2 Waymo Open Dataset

Waymo Open [Sun et al., 2020] provides high-quality autonomous driving data:

- **Data:** 1150 scenes, 20 seconds each
- **Sensors:** 5 lidars, 5 cameras
- **Quality:** Higher annotation quality than nuScenes
- **Challenges:** 3D detection, tracking, motion prediction

1.4.3 Argoverse 2

Argoverse 2 [Wilson et al., 2023] emphasizes HD maps and motion forecasting:

- **Data:** 1000 sequences with HD maps
- **Focus:** Motion forecasting, 3D object detection
- **Maps:** High-definition vector maps with lane-level information

2 Detailed Method Descriptions

This appendix provides detailed technical descriptions of key methods referenced in the main report.

2.1 ReAct: Synergizing Reasoning and Acting

ReAct [Yao et al., 2023] interleaves reasoning traces with actions:
Algorithm:

1. Given task description, generate thought about what to do
2. Based on thought, select and execute action
3. Observe result of action
4. Generate next thought based on observation
5. Repeat until task complete or max steps reached

Key Insight: By making reasoning explicit, the model can better plan and recover from errors. The thought traces also provide interpretability.

Limitations: ReAct relies on the LLM's ability to generate useful thoughts. For complex spatial tasks, the reasoning may be superficial or incorrect.

2.2 Reflexion: Language Agents with Verbal Reinforcement Learning

Reflexion [Shinn et al., 2023] adds self-reflection to improve over trials:

Components:

- **Actor:** Executes actions in environment
- **Evaluator:** Assesses task success
- **Self-Reflection:** Generates verbal feedback on failures
- **Memory:** Stores reflections for future trials

Process:

1. Actor attempts task
2. Evaluator determines success/failure
3. If failure, Self-Reflection generates analysis
4. Reflection stored in Memory
5. Actor retries with reflection as additional context

2.3 DreamerV3: Mastering Diverse Domains through World Models

DreamerV3 [Hafner et al., 2023] learns a world model for planning:

World Model Components:

- **Encoder:** Maps observations to latent states
- **Dynamics:** Predicts next latent state given action
- **Decoder:** Reconstructs observations from latent states
- **Reward Predictor:** Predicts rewards from latent states

Training:

1. Collect experience in environment
2. Train world model on collected data
3. Imagine trajectories using world model
4. Train actor-critic on imagined trajectories

Key Innovations:

- Symlog predictions for numerical stability
- Fixed hyperparameters across domains
- Discrete latent representations

2.4 RT-2: Vision-Language-Action Models

RT-2 [Brohan et al., 2023] treats robot actions as language tokens:

Architecture:

- Base: PaLI-X (55B) or PaLM-E (12B)
- Input: Image + text instruction
- Output: Action tokens (discretized robot commands)

Action Tokenization:

- Discretize continuous actions into 256 bins
- Represent as text tokens (e.g., "1 128 64 32 ...")
- Decode tokens back to continuous actions

Emergent Capabilities:

- Symbol understanding (move to X on table)
- Reasoning (pick up object that doesn't belong)
- Multi-step planning

2.5 VLMaps: Visual Language Maps for Robot Navigation

VLMaps [Huang et al., 2023a] creates language-indexed spatial maps:

Map Construction:

1. Robot explores environment with RGB-D camera
2. Extract CLIP features for each pixel
3. Project features to 3D voxel grid
4. Aggregate features across viewpoints

Querying:

1. Encode natural language query with CLIP text encoder
2. Compute similarity with map features
3. Return locations with highest similarity

Applications:

- "Where is the refrigerator?" $\rightarrow$ Returns location
- "Navigate to the couch" $\rightarrow$ Plans path to couch location

3 Implementation Recipes

This appendix provides practical implementation guidance for common spatial AI tasks.

3.1 Recipe 1: Building a RAG-Enhanced Spatial Agent

Step 1: Set Up Vector Database

```
import chromadb
from sentence_transformers import SentenceTransformer

# Initialize embedding model
embedder = SentenceTransformer('all-MiniLM-L6-v2')

# Initialize ChromaDB
client = chromadb.Client()
collection = client.create_collection(
    name="spatial_knowledge",
    metadata={"hnsw:space": "cosine"}
)
```

Step 2: Index Spatial Knowledge

```
def index_spatial_document(doc_id, text, location):
    embedding = embedder.encode(text)
    collection.add(
        ids=[doc_id],
        embeddings=[embedding.tolist()],
        metadatas=[{"location": location}],
        documents=[text]
    )
```

Step 3: Retrieve Relevant Context

```
def retrieve_context(query, n_results=5):
    query_embedding = embedder.encode(query)
    results = collection.query(
        query_embeddings=[query_embedding.tolist()],
        n_results=n_results
    )
    return results['documents'][0]
```

Step 4: Generate Response with Context

```
def generate_response(query, context):
    prompt = f"""Context: {context}

    Question: {query}

    Answer based on the context:"""

    response = llm.generate(prompt)
    return response
```

3.2 Recipe 2: Training a GNN for Network Failure Prediction

Step 1: Prepare Graph Data

```
import torch
from torch_geometric.data import Data

def prepare_network_graph(nodes, edges, features, labels):
    # Node features: [capacity, age, maintenance_count, ...]
    x = torch.tensor(features, dtype=torch.float)

```

```
    # Edge index: [2, num_edges]
    edge_index = torch.tensor(edges, dtype=torch.long).t()

    # Labels: binary failure indicator
    y = torch.tensor(labels, dtype=torch.long)

    return Data(x=x, edge_index=edge_index, y=y)
```

Step 2: Define GNN Model

```
import torch.nn.functional as F
from torch_geometric.nn import GCNConv

class FailurePredictionGNN(torch.nn.Module):
    def __init__(self, in_channels, hidden_channels):
        super().__init__()
        self.conv1 = GCNConv(in_channels, hidden_channels)
        self.conv2 = GCNConv(hidden_channels, hidden_channels)
        self.classifier = torch.nn.Linear(hidden_channels, 2)

    def forward(self, x, edge_index):
        x = self.conv1(x, edge_index)
        x = F.relu(x)
        x = F.dropout(x, p=0.5, training=self.training)
        x = self.conv2(x, edge_index)
        x = self.classifier(x)
        return F.log_softmax(x, dim=1)
```

Step 3: Training Loop

```
def train(model, data, optimizer):
    model.train()
    optimizer.zero_grad()
    out = model(data.x, data.edge_index)
    loss = F.nll_loss(out[data.train_mask], data.y[data.train_mask])
    loss.backward()
    optimizer.step()
    return loss.item()

def evaluate(model, data):
    model.eval()
    with torch.no_grad():
        out = model(data.x, data.edge_index)
        pred = out.argmax(dim=1)
        correct = (pred[data.test_mask] == data.y[data.test_mask]).sum()
        acc = correct / data.test_mask.sum()
    return acc.item()
```

3.3 Recipe 3: Building an MCP Tool Server

Step 1: Define Tool Schema

```
TOOL_SCHEMA = {
    "name": "analyze_network_segment",
    "description": "Analyzes a network segment for capacity and risk",
    "parameters": {
        "type": "object",
        "properties": {
            "segment_id": {
                "type": "string",
```

```
                "description": "Unique identifier for the network segment"
            },
            "analysis_type": {
                "type": "string",
                "enum": ["capacity", "risk", "both"],
                "description": "Type of analysis to perform"
            }
        },
        "required": ["segment_id"]
    }
}
```

Step 2: Implement Tool Handler

```
async def handle_analyze_network_segment(params):
    segment_id = params["segment_id"]
    analysis_type = params.get("analysis_type", "both")

    # Fetch segment data from database
    segment = await db.get_segment(segment_id)

    result = {}

    if analysis_type in ["capacity", "both"]:
        result["capacity"] = {
            "current_utilization": segment.current_load / segment.capacity,
            "peak_utilization": segment.peak_load / segment.capacity,
            "headroom": segment.capacity - segment.current_load
        }

    if analysis_type in ["risk", "both"]:
        # Run GNN model for risk prediction
        risk_score = await gnn_model.predict_risk(segment_id)
        result["risk"] = {
            "failure_probability": risk_score,
            "risk_factors": await get_risk_factors(segment_id)
        }

    return result
```

4 Comprehensive Literature Tables

This appendix provides comprehensive tables of key papers organized by topic.

4.1 Agentic AI Methods

Table 11: Key Agentic AI Methods

Method	Year	Venue	Key Contribution
ReAct	2023	ICLR	Interleaved reasoning and acting
Reflexion	2023	NeurIPS	Self-reflection for improvement
Toolformer	2023	NeurIPS	Self-supervised tool learning

Method	Year	Venue	Key Contribution
AutoGPT	2023	-	Autonomous goal pursuit
Voyager	2023	NeurIPS	Open-ended exploration in Minecraft
MemGPT	2023	-	Hierarchical memory management
AutoGen	2023	-	Multi-agent conversation framework
MetaGPT	2023	ICLR	Role-based multi-agent collaboration
CAMEL	2023	NeurIPS	Role-playing for agent collaboration
AgentVerse	2024	-	Multi-agent simulation platform

4.2 Vision-Language-Action Models

Table 12: Key Vision-Language-Action Models

Model	Year	Venue	Key Contribution
RT-1	2022	RSS	Robotics Transformer architecture
RT-2	2023	CoRL	VLM backbone for robot control
PaLM-E	2023	ICML	Embodied multimodal language model
Octo	2024	RSS	Open-source generalist robot policy
OpenVLA	2024	CoRL	Open-source VLA with strong performance
π_0	2024	-	Physical Intelligence foundation model
GR-1	2024	-	Generalist robot with world model
RVT-2	2024	CoRL	Efficient multi-view transformer

4.3 Graph Neural Networks for Spatial Data

Table 13: Key GNN Methods for Spatial Data

Method	Year	Venue	Key Contribution
GCN	2017	ICLR	Spectral graph convolutions
GAT	2018	ICLR	Attention-based aggregation
GraphSAGE	2017	NeurIPS	Inductive learning on graphs
DCRNN	2018	ICLR	Diffusion convolution for traffic

Method	Year	Venue	Key Contribution
STGCN	2018	IJCAI	Spatio-temporal graph convolution
Graph WaveNet	2019	IJCAI	Adaptive adjacency learning
GNN-RAG	2024	-	GNN for knowledge graph retrieval
GraphGPT	2024	-	Graph-language model alignment

4.4 World Models

Table 14: Key World Model Methods

Method	Year	Venue	Key Contribution
World Models	2018	NeurIPS	VAE + MDN-RNN for imagination
Dreamer	2020	ICLR	Latent imagination for control
DreamerV2	2021	ICLR	Discrete latents, Atari mastery
DreamerV3	2023	ICLR	Universal world model
DayDreamer	2023	CoRL	Real robot world model transfer
Genie	2024	ICML	Controllable video world model
GAIA-1	2023	-	Driving video world model
Sora	2024	-	Video generation as world simulation

5 Glossary of Terms

Agentic AI
AI systems that can autonomously perceive, reason, and act to achieve goals.

BEV (Bird's Eye View)
A top-down representation of a scene, commonly used in autonomous driving.

Chain-of-Thought (CoT)
A prompting technique that elicits step-by-step reasoning from LLMs.

Cognitive Map
A mental representation of spatial relationships in an environment.

Embodied AI
AI systems that interact with the physical world through sensors and actuators.

Foundation Model
A large model trained on broad data that can be adapted to many downstream tasks.

GNN (Graph Neural Network)
Neural networks designed to operate on graph-structured data.

Grounding
Connecting abstract concepts (e.g., language) to concrete entities (e.g., objects, locations).

LLM (Large Language Model)
Neural networks trained on large text corpora for language understanding and generation.

MCP (Model Context Protocol)
A standardized interface for LLM agents to interact with external tools.

MLLM (Multimodal Large Language Model)
LLMs that can process multiple modalities (text, images, etc.).

NeRF (Neural Radiance Field)
A neural representation for novel view synthesis.

RAG (Retrieval-Augmented Generation)
Enhancing LLM generation with retrieved external knowledge.

ReAct
A framework for interleaving reasoning and acting in LLM agents.

Reflexion
A framework for LLM agents to learn from self-reflection.

Sim-to-Real
Transferring policies learned in simulation to the real world.

Spatial Intelligence
The ability to perceive, reason about, and interact with 3D environments.

SPL (Success weighted by Path Length)
A navigation metric that rewards both success and efficiency.

TAMP (Task and Motion Planning)
Planning that combines symbolic task planning with geometric motion planning.

VLA (Vision-Language-Action)
Models that map visual and language inputs to robot actions.

VLM (Vision-Language Model)
Models that jointly process visual and textual information.

VLN (Vision-Language Navigation)
Navigation guided by natural language instructions.

World Model
A learned model of environment dynamics used for planning.

6 Extended Competitive Analysis

This appendix provides extended analysis of the competitive landscape.

6.1 Detailed Company Profiles

6.1.1 Esri (ArcGIS)

Overview: Esri is the dominant player in the GIS market with over 40% market share. ArcGIS is the industry standard for geospatial analysis.

Strengths:

- Comprehensive GIS platform with decades of development
- Strong enterprise relationships and brand recognition
- Extensive ecosystem of partners and developers
- GeoAI capabilities being added

Weaknesses:

- Legacy architecture not designed for AI-native workflows
- High licensing costs
- Steep learning curve
- AI features are add-ons, not core architecture

AtlasPro Differentiation: Our AI-native architecture enables agentic workflows that are fundamentally impossible with ArcGIS's tool-based paradigm. We can automate complex analysis that would require manual configuration in ArcGIS.

6.1.2 IQGeo (Comsof Fiber)

Overview: IQGeo acquired Comsof in 2021, gaining their fiber network planning software. Comsof Fiber is the leading automated fiber planning tool.

Strengths:

- Purpose-built for fiber network planning
- Automated design generation
- Strong customer base in telecom
- Integration with major GIS platforms

Weaknesses:

- Heuristic-based optimization, not machine learning
- Cannot learn from historical data
- Limited predictive capabilities
- No AI agent integration

AtlasPro Differentiation: Our GNN-based approach enables predictive analytics (failure prediction, demand forecasting) that are fundamentally impossible with Comsof's rule-based system. Our agentic interface enables natural language interaction.

6.1.3 World Labs

Overview: Founded by Fei-Fei Li in 2024, World Labs is building frontier spatial AI models. Raised $230M at $1B+ valuation.

Strengths:

- World-class research team
- Significant funding
- Frontier model capabilities
- Strong academic connections

Weaknesses:

- Focus on consumer/creative applications, not B2B infrastructure
- No domain expertise in telecom/utilities
- Early stage, no production deployments

AtlasPro Differentiation: World Labs is building horizontal spatial AI capabilities. AtlasPro is building a vertical solution for network infrastructure. Our domain expertise and customer relationships create a advantage that World Labs would need years to replicate.

6.2 Market Opportunity Sizing

Total Addressable Market (TAM):

- Global geospatial analytics: $150B by 2030
- AI-powered segment: $50B by 2030

Serviceable Addressable Market (SAM):

- Telecom network planning and management: $8B
- Utility network intelligence: $5B
- Total SAM: $13B

Serviceable Obtainable Market (SOM):

- Initial target: North American fiber ISPs
- Market size: $500M
- 5-year target: 5% market share = $25M ARR

7 Risk Analysis and Mitigation

This appendix provides detailed analysis of risks facing AtlasPro AI and mitigation strategies.

7.1 Technical Risks

Risk 1: GNN Model Performance

- **Description:** GNN models may not achieve sufficient accuracy for production use
- **Likelihood:** Medium
- **Impact:** High
- **Mitigation:** Start with simpler models, iterate based on customer feedback, maintain fallback to rule-based systems

Risk 2: Data Quality

- **Description:** Customer data may be incomplete, inconsistent, or inaccurate
- **Likelihood:** High
- **Impact:** Medium
- **Mitigation:** Build robust data validation pipelines, develop data quality metrics, provide data cleaning tools

Risk 3: LLM Reliability

- **Description:** LLM agents may generate incorrect or harmful outputs
- **Likelihood:** Medium
- **Impact:** High
- **Mitigation:** Implement output validation, human-in-the-loop for critical decisions, comprehensive testing

7.2 Market Risks

Risk 4: Incumbent Response

- **Description:** Esri or IQGeo may develop competing AI capabilities
- **Likelihood:** High
- **Impact:** Medium
- **Mitigation:** Move fast, build customer relationships, create switching costs through data integration

Risk 5: Customer Adoption

- **Description:** Customers may be slow to adopt AI-based solutions
- **Likelihood:** Medium
- **Impact:** High
- **Mitigation:** Start with low-risk use cases, demonstrate clear ROI, provide extensive training and support

7.3 Operational Risks

Risk 6: Talent Acquisition

- **Description:** Difficulty hiring qualified ML engineers and domain experts
- **Likelihood:** High
- **Impact:** Medium
- **Mitigation:** Competitive compensation, remote-friendly culture, strong research reputation

Risk 7: Scaling Challenges

- **Description:** Infrastructure may not scale to handle large customer deployments
- **Likelihood:** Medium
- **Impact:** High
- **Mitigation:** Cloud-native architecture, load testing, gradual rollout

8 Evaluation Metrics and KPIs

This appendix defines the key metrics for evaluating AtlasPro AI's technical and business performance.

8.1 Model Performance Metrics

Failure Prediction:

- Precision at 90-day horizon
- Recall at 90-day horizon
- F1 score
- AUC-ROC

Capacity Forecasting:

- Mean Absolute Percentage Error (MAPE)
- Root Mean Square Error (RMSE)
- Forecast horizon accuracy (7-day, 30-day, 90-day)

Routing Optimization:

- Cost reduction vs. baseline
- Constraint satisfaction rate
- Computation time

8.2 Agent Performance Metrics

Task Completion:

- Success rate on benchmark queries
- Average steps to completion
- Error rate

User Satisfaction:

- Query response time
- Answer accuracy (human evaluation)
- User feedback scores

8.3 Business Metrics

Customer Metrics:

- Number of active customers
- Customer retention rate
- Net Promoter Score (NPS)
- Customer lifetime value (CLV)

Financial Metrics:

- Annual Recurring Revenue (ARR)
- Monthly Recurring Revenue (MRR)
- Gross margin
- Customer acquisition cost (CAC)

9 Case Studies and Use Case Analysis

This appendix provides detailed case studies demonstrating AtlasPro AI's approach to real-world problems.

9.1 Case Study 1: Fiber Network Failure Prediction

9.1.1 Problem Statement

A mid-sized fiber ISP with 50,000 miles of fiber network experiences approximately 200 unplanned outages per year. Each outage costs an average of $15,000 in direct repair costs plus $50,000 in customer churn and SLA penalties. The total annual cost of unplanned outages exceeds $13 million.

9.1.2 Current Approach

The ISP currently uses a reactive maintenance approach:

- Wait for customer complaints or monitoring alerts
- Dispatch technicians to diagnose and repair
- No systematic analysis of failure patterns
- Maintenance schedules based on manufacturer recommendations, not actual conditions

9.1.3 AtlasPro AI Solution

Our approach uses a spatio-temporal GNN to predict failures before they occur:

Data Integration:

- Network topology from GIS (nodes, edges, equipment types)
- Historical maintenance records (5 years)
- Weather data (temperature, precipitation, wind)
- Traffic patterns (utilization over time)
- Equipment age and specifications

Model Architecture:

- Graph representation: Each splice, pole, and equipment as node
- Node features: Age, type, maintenance history, environmental exposure
- Edge features: Cable type, length, burial depth, terrain
- Temporal features: 90-day rolling window of utilization and weather

Training Approach:

- Binary classification: Will this component fail in next 90 days?
- Class imbalance handling: Focal loss, oversampling
- Validation: Time-based split to prevent data leakage

9.1.4 Expected Results

Based on our analysis of similar deployments in the literature:

- Precision: 70-80% (of predicted failures, 70-80% actually occur)
- Recall: 50-60% (of actual failures, 50-60% are predicted)
- Cost reduction: 30-40% reduction in outage-related costs
- ROI: 5-10x return on investment in first year

Table 15: Failure Prediction Implementation Timeline

Phase	Duration	Activities
Data Integration	4 weeks	Connect to GIS, import historical data
Model Development	6 weeks	Train and validate GNN model
Pilot Deployment	8 weeks	Deploy to subset of network, validate predictions
Full Rollout	4 weeks	Extend to entire network

9.1.5 Implementation Timeline

9.2 Case Study 2: Intelligent Network Planning Assistant

9.2.1 Problem Statement

A regional utility company is planning a major grid modernization project. The planning team needs to evaluate hundreds of potential configurations, considering factors like:

- Load growth projections
- Renewable energy integration
- Reliability requirements
- Budget constraints
- Regulatory compliance

Traditional planning tools require manual configuration of each scenario, taking weeks to evaluate alternatives.

9.2.2 AtlasPro AI Solution

Our agentic planning assistant enables natural language interaction:

User Query: "Show me the top 3 options for adding 50MW of solar capacity to the western region while maintaining N-1 reliability and staying under $20M budget."

Agent Workflow:

1. Parse query to extract constraints (capacity, region, reliability, budget)
2. Retrieve relevant network data from graph database
3. Generate candidate configurations using optimization model
4. Evaluate each configuration against constraints
5. Rank by multi-objective score (cost, reliability, future flexibility)
6. Present results with explanations

Key Capabilities:

- Natural language understanding of planning requirements
- Automatic constraint extraction and validation
- Multi-objective optimization with explainable trade-offs
- Interactive refinement based on user feedback

9.2.3 Differentiation from Traditional Tools

Table 16: Planning Assistant vs. Traditional Tools

Aspect	Traditional Tools	AtlasPro AI
Interface	GUI with manual configuration	Natural language
Scenario Generation	Manual	Automated
Constraint Handling	Hard-coded	Flexible, learned
Explanation	Limited	Full reasoning trace
Iteration Time	Hours to days	Minutes

9.3 Case Study 3: Real-Time Network Monitoring and Anomaly Detection

9.3.1 Problem Statement

A large telecommunications provider operates a nationwide fiber network with millions of endpoints. Current monitoring systems generate thousands of alerts per day, overwhelming the network operations center (NOC). Most alerts are false positives or low-priority issues.

9.3.2 AtlasPro AI Solution

Our approach uses GNN-based anomaly detection to prioritize alerts:

Architecture:

- Real-time ingestion of network telemetry (latency, packet loss, utilization)
- GNN learns normal behavior patterns for each network segment
- Anomalies detected as deviations from learned patterns
- Context-aware prioritization based on impact analysis

Key Features:

- **Spatial Context:** Anomaly in backbone segment prioritized over edge segment
- **Temporal Context:** Anomaly during peak hours prioritized over off-peak
- **Correlation:** Related anomalies grouped to identify root cause
- **Impact Estimation:** Number of affected customers calculated

9.3.3 Expected Results

- Alert reduction: 80-90% reduction in false positives
- MTTR improvement: 30-50% reduction in mean time to resolution
- NOC efficiency: 2-3x increase in issues resolved per analyst

10 Mathematical Foundations

This appendix provides detailed mathematical foundations for the methods discussed in this report.

10.1 Graph Neural Network Fundamentals

10.1.1 Graph Representation

A graph $G = (V, E)$ consists of:

- V: Set of n nodes (vertices)
- $E \subseteq V \times V$: Set of edges
- $\mathbf{X} \in \mathbb{R}^{n \times d}$: Node feature matrix
- $\mathbf{A} \in \{0, 1\}^{n \times n}$: Adjacency matrix

10.1.2 Message Passing Framework

GNNs operate through message passing [Gilmer et al., 2017]:

$$\mathbf{h}_v^{(k)} = \text{UPDATE}^{(k)}\left(\mathbf{h}_v^{(k-1)}, \text{AGGREGATE}^{(k)}\left(\{\mathbf{h}_u^{(k-1)} : u \in \mathcal{N}(v)\}\right)\right) \tag{3}$$

where:

- $\mathbf{h}_v^{(k)}$: Hidden state of node v at layer k
- $\mathcal{N}(v)$: Neighbors of node v
- AGGREGATE: Permutation-invariant function (sum, mean, max)
- UPDATE: Learnable transformation (MLP)

10.1.3 Graph Convolutional Network (GCN)

GCN [Kipf and Welling, 2017] uses spectral convolutions:

$$\mathbf{H}^{(k)} = \sigma\left(\tilde{\mathbf{D}}^{-1/2}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-1/2}\mathbf{H}^{(k-1)}\mathbf{W}^{(k)}\right) \tag{4}$$

where:

- $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$: Adjacency with self-loops
- $\tilde{\mathbf{D}}_{ii} = \sum_j \tilde{\mathbf{A}}_{ij}$: Degree matrix
- $\mathbf{W}^{(k)}$: Learnable weight matrix
- σ: Activation function (ReLU)

10.1.4 Graph Attention Network (GAT)

GAT [Velickovic et al., 2018] uses attention mechanisms:

$$\mathbf{h}_v^{(k)} = \sigma\left(\sum_{u \in \mathcal{N}(v) \cup \{v\}} \alpha_{vu} \mathbf{W}^{(k)} \mathbf{h}_u^{(k-1)}\right) \tag{5}$$

where attention coefficients are computed as:

$$\alpha_{vu} = \frac{\exp\left(\text{LeakyReLU}\left(\mathbf{a}^T[\mathbf{W}\mathbf{h}_v \| \mathbf{W}\mathbf{h}_u]\right)\right)}{\sum_{w \in \mathcal{N}(v)} \exp\left(\text{LeakyReLU}\left(\mathbf{a}^T[\mathbf{W}\mathbf{h}_v \| \mathbf{W}\mathbf{h}_w]\right)\right)} \tag{6}$$

10.2 World Model Mathematics

10.2.1 Latent Dynamics Model

World models learn a latent dynamics model [Hafner et al., 2019]:
Encoder: $q_\phi(z_t|o_t)$ maps observations to latent states
Dynamics: $p_\theta(z_{t+1}|z_t, a_t)$ predicts next latent state
Decoder: $p_\theta(o_t|z_t)$ reconstructs observations
Reward: $p_\theta(r_t|z_t)$ predicts rewards

10.2.2 Training Objective

The model is trained to maximize the evidence lower bound (ELBO):

$$\mathcal{L} = \mathbb{E}_{q_\phi}\left[\sum_{t=1}^{T} \log p_\theta(o_t|z_t) + \log p_\theta(r_t|z_t) - \beta \text{KL}[q_\phi(z_t|o_t) \| p_\theta(z_t|z_{t-1}, a_{t-1})]\right] \tag{7}$$

10.2.3 Planning in Imagination

Once trained, the world model enables planning without environment interaction:

1. Sample initial latent state from encoder
2. Imagine trajectories using dynamics model
3. Evaluate trajectories using reward model
4. Select action sequence with highest expected return

10.3 Reinforcement Learning Foundations

10.3.1 Markov Decision Process

An MDP is defined by $(\mathcal{S}, \mathcal{A}, P, R, \gamma)$:

- $\mathcal{S}$: State space
- $\mathcal{A}$: Action space
- $P(s'|s, a)$: Transition probability

- $R(s, a)$: Reward function
- $\gamma \in [0, 1)$: Discount factor

10.3.2 Value Functions

State Value:

$$V^{\pi}(s) = \mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) | s_0 = s\right] \tag{8}$$

Action Value:

$$Q^{\pi}(s, a) = \mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) | s_0 = s, a_0 = a\right] \tag{9}$$

10.3.3 Policy Gradient

Policy gradient methods optimize the policy directly:

$$\nabla_{\theta} J(\theta) = \mathbb{E}_{\pi_{\theta}}\left[\nabla_{\theta} \log \pi_{\theta}(a|s) Q^{\pi_{\theta}}(s, a)\right] \tag{10}$$

11 Extended Technical Specifications

This appendix provides detailed technical specifications for AtlasPro AI's proposed architecture.

11.1 System Architecture

11.1.1 High-Level Components

1. **Data Layer**
    - Graph database (Neo4j or similar) for network topology
    - Time-series database (InfluxDB or similar) for telemetry
    - Vector database (Pinecone or similar) for embeddings
    - Object storage (S3) for large files and models
2. **Model Layer**
    - GNN models for network analysis
    - World models for simulation
    - LLM for natural language understanding
    - Embedding models for retrieval
3. **Agent Layer**
    - MCP server exposing tools
    - Agent orchestrator (ReAct-style)
    - Memory management (short-term, long-term, episodic)
    - Safety guardrails and output validation
4. **Application Layer**

- Web interface for interactive queries
- API for programmatic access
- Dashboard for monitoring and analytics
- Integration connectors for existing systems

11.1.2 Deployment Architecture

- **Cloud Platform:** AWS, GCP, or Azure
- **Container Orchestration:** Kubernetes
- **Model Serving:** NVIDIA Triton or similar
- **API Gateway:** Kong or AWS API Gateway
- **Monitoring:** Prometheus + Grafana

11.2 Data Schema

11.2.1 Node Types

```python
@dataclass
class NetworkNode:
    id: str
    type: Literal["splice", "pole", "cabinet", "substation", "customer"]
    location: Point  # (lat, lon)
    properties: Dict[str, Any]
    created_at: datetime
    updated_at: datetime

@dataclass
class SpliceNode(NetworkNode):
    splice_type: str
    fiber_count: int
    enclosure_type: str
    installation_date: date
    last_maintenance: date

@dataclass
class PoleNode(NetworkNode):
    height_meters: float
    material: Literal["wood", "steel", "concrete"]
    owner: str
    attachments: List[str]
```

Listing 5: Node Schema

11.2.2 Edge Types

```python
@dataclass
class NetworkEdge:
    id: str
    source_id: str
    target_id: str
    type: Literal["fiber", "copper", "wireless"]
```

```
    properties: Dict[str, Any]

@dataclass
class FiberEdge(NetworkEdge):
    fiber_count: int
    cable_type: str
    length_meters: float
    installation_date: date
    burial_depth_meters: Optional[float]
    aerial: bool
    capacity_gbps: float
    current_utilization: float
```

Listing 6: Edge Schema

11.3 API Specifications

11.3.1 REST API Endpoints

```
# Network Analysis
GET  /api/v1/network/{network_id}/topology
GET  /api/v1/network/{network_id}/nodes
GET  /api/v1/network/{network_id}/edges
POST /api/v1/network/{network_id}/analyze

# Predictions
POST /api/v1/predict/failure
POST /api/v1/predict/capacity
POST /api/v1/predict/demand

# Planning
POST /api/v1/plan/route
POST /api/v1/plan/optimize
POST /api/v1/plan/simulate

# Agent
POST /api/v1/agent/query
GET  /api/v1/agent/session/{session_id}
POST /api/v1/agent/feedback
```

Listing 7: API Endpoints

11.3.2 MCP Tool Definitions

```
TOOLS = [
    {
        "name": "get_network_topology",
        "description": "Retrieves the network topology for a specified area",
        "parameters": {
            "type": "object",
            "properties": {
                "area": {"type": "geojson"},
                "include_equipment": {"type": "boolean", "default": True},
                "max_depth": {"type": "integer", "default": 3}
            },
            "required": ["area"]
        }
```

```
    },
    {
        "name": "analyze_capacity",
        "description": "Analyzes capacity utilization for network segments",
        "parameters": {
            "type": "object",
            "properties": {
                "segment_ids": {"type": "array", "items": {"type": "string"}},
                "time_range": {"type": "string", "enum": ["1h", "24h", "7d", "30d"]},
                "metrics": {"type": "array", "items": {"type": "string"}}
            },
            "required": ["segment_ids"]
        }
    },
    {
        "name": "predict_failures",
        "description": "Predicts component failures using GNN model",
        "parameters": {
            "type": "object",
            "properties": {
                "component_type": {"type": "string"},
                "area": {"type": "geojson"},
                "horizon_days": {"type": "integer", "default": 90},
                "threshold": {"type": "number", "default": 0.5}
            },
            "required": []
        }
    },
    {
        "name": "optimize_route",
        "description": "Generates optimal routing for new connections",
        "parameters": {
            "type": "object",
            "properties": {
                "source": {"type": "geojson"},
                "destination": {"type": "geojson"},
                "constraints": {"type": "object"},
                "objectives": {"type": "array", "items": {"type": "string"}}
            },
            "required": ["source", "destination"]
        }
    },
    {
        "name": "simulate_change",
        "description": "Simulates the impact of a network change",
        "parameters": {
            "type": "object",
            "properties": {
                "change_type": {"type": "string", "enum": ["add", "remove", "modify"]},
                "target_ids": {"type": "array", "items": {"type": "string"}},
                "parameters": {"type": "object"}
            },
            "required": ["change_type", "target_ids"]
        }
    }
```

```
]
```

Listing 8: Complete MCP Tool Schema

12 Survey of Related Work

This appendix provides an extended survey of related work across key research areas.

12.1 Large Language Models for Agents

The emergence of capable LLMs has enabled a new paradigm of AI agents that can reason, plan, and act autonomously.

Foundation Models. GPT-4 [OpenAI, 2023] demonstrated emergent capabilities in reasoning, planning, and tool use. Claude [Anthropic, 2024] introduced constitutional AI for safer agent behavior. Gemini [Team et al., 2023] achieved strong multimodal understanding.

Agent Frameworks. LangChain and LlamaIndex provide infrastructure for building LLM agents. AutoGen [Wu et al., 2023] enables multi-agent conversations. CrewAI focuses on role-based agent collaboration.

Tool Use. Toolformer [Schick et al., 2023] showed LLMs can learn to use tools through self-supervision. Gorilla [Patil et al., 2023] specialized in API calling. ToolBench [Qin et al., 2023] provides comprehensive tool use evaluation.

12.2 Embodied AI and Robotics

Embodied AI focuses on agents that interact with the physical world.

Simulation Platforms. Habitat [Savva et al., 2019] provides photorealistic indoor simulation. Isaac Sim enables high-fidelity robot simulation. AI2-THOR focuses on interactive household environments.

Robot Learning. RT-1 [Brohan et al., 2022] introduced transformer architectures for robot control. RT-2 [Brohan et al., 2023] leveraged VLM pretraining. Octo [Team et al., 2024] provided an open-source generalist policy.

Manipulation. PerAct [Shridhar et al., 2023] used 3D voxel representations. RVT [Goyal et al., 2023] introduced efficient multi-view transformers. VoxPoser [Huang et al., 2023b] enabled zero-shot manipulation through LLM-generated affordances.

12.3 Geospatial AI

Geospatial AI applies machine learning to geographic and spatial data.

Remote Sensing. Prithvi [Jakubik et al., 2024] is NASA/IBM's geospatial foundation model. SatMAE [Cong et al., 2022] applies masked autoencoders to satellite imagery. Clay provides open-source earth observation models.

Urban Computing. Traffic prediction has been revolutionized by GNNs [Li et al., 2018, Yu et al., 2018]. Urban flow prediction enables smart city applications. Location-based services leverage spatial embeddings.

GeoAI Platforms. Esri's GeoAI tools integrate ML with ArcGIS. Google Earth Engine provides planetary-scale analysis. Microsoft's Planetary Computer offers open geospatial data.

12.4 Graph Neural Networks

GNNs have become the standard approach for learning on graph-structured data.

Foundational Methods. GCN [Kipf and Welling, 2017] introduced spectral convolutions. GAT [Velickovic et al., 2018] added attention mechanisms. GraphSAGE [Hamilton et al., 2017] enabled inductive learning.

Spatio-Temporal GNNs. DCRNN [Li et al., 2018] combined diffusion convolution with RNNs. STGCN [Yu et al., 2018] used temporal convolutions. Graph WaveNet [Wu et al., 2019] learned adaptive adjacency matrices.

GNN-LLM Integration. GraphGPT [Tang et al., 2024] aligned graph encoders with LLMs. GNN-RAG [Wang et al., 2024a] used GNNs for knowledge graph retrieval. LLaGA explored instruction tuning for graph tasks.

13 Detailed Experimental Protocols

This appendix describes the experimental protocols we plan to use for validating AtlasPro AI's approach.

13.1 Failure Prediction Evaluation

13.1.1 Dataset Preparation

1. **Data Collection:** Obtain 5+ years of maintenance records from partner ISPs
2. **Label Definition:** Component failure = unplanned outage requiring repair
3. **Feature Engineering:** Extract node and edge features from GIS and telemetry
4. **Train/Val/Test Split:** Time-based split (e.g., train on 2019-2023, test on 2024)

13.1.2 Evaluation Metrics

- **Precision@k:** Of top k predicted failures, how many actually occurred?
- **Recall@k:** Of actual failures, how many were in top k predictions?
- **AUC-ROC:** Area under receiver operating characteristic curve
- **Calibration:** Are predicted probabilities well-calibrated?

13.1.3 Baselines

1. **Age-based:** Predict failure based on component age alone
2. **Random Forest:** Traditional ML on tabular features
3. **MLP:** Neural network on tabular features (no graph structure)
4. **GCN:** Standard graph convolutional network
5. **GAT:** Graph attention network
6. **AtlasPro GNN:** Our spatio-temporal architecture

13.2 Agent Evaluation

13.2.1 Benchmark Tasks

We define a benchmark suite of 50 representative tasks:

1. **Information Retrieval (10 tasks):** "What is the capacity of segment X?"
2. **Analysis (15 tasks):** "Find all segments over 80% capacity in region Y"
3. **Prediction (10 tasks):** "Which components are most likely to fail next quarter?"
4. **Planning (10 tasks):** "Design a route from A to B minimizing cost"
5. **Multi-step (5 tasks):** "Find high-risk, high-capacity segments and suggest rerouting"

13.2.2 Evaluation Criteria

- **Correctness:** Does the answer match ground truth?
- **Completeness:** Does the answer address all aspects of the query?
- **Efficiency:** How many steps/tokens were required?
- **Safety:** Did the agent avoid harmful actions?

13.2.3 Human Evaluation

For subjective quality assessment:

1. Recruit domain experts (network engineers, planners)
2. Present agent responses alongside baseline responses
3. Rate on 5-point scale for helpfulness, accuracy, clarity
4. Compute inter-rater agreement (Krippendorff's alpha)

14 Future Research Directions

This appendix outlines promising research directions beyond the scope of this report.

14.1 Multimodal Spatial Understanding

Current approaches primarily use structured data (graphs, coordinates). Future work should integrate:

- Satellite and aerial imagery for visual context
- Street-level imagery for detailed inspection
- LiDAR point clouds for 3D understanding
- Document understanding for permits, specifications

14.2 Federated Learning for Privacy

Network data is sensitive. Federated learning could enable:

- Training on distributed customer data without centralization
- Privacy-preserving model updates
- Collaborative improvement across organizations

14.3 Causal Reasoning for Intervention Planning

Current ML models are correlational. Causal methods could enable:

- Understanding why failures occur, not just predicting them
- Estimating intervention effects before deployment
- Counterfactual analysis ("What if we had done X?")

14.4 Continuous Learning and Adaptation

Networks evolve over time. Continuous learning could enable:

- Automatic model updates as new data arrives
- Adaptation to distribution shift (new equipment, changing patterns)
- Lifelong learning without catastrophic forgetting

14.5 Human-AI Collaboration

Optimal systems combine AI capabilities with human expertise:

- Interactive refinement of AI recommendations
- Explanation and justification for trust building
- Graceful degradation when AI is uncertain
- Learning from human corrections

15 Acknowledgments and Contributions

15.1 Author Contributions

- **Gloria Felicia:** Project lead, research direction, writing
- **Nolan Bryant:** GNN architecture, implementation
- **Handi Putra:** World models, simulation
- **Ayaan Gazali:** Agent framework, MCP integration
- **Eliel Lobo:** Competitive analysis, market research
- **Esteban Rojas:** Data infrastructure, benchmarking

15.2 Acknowledgments

We thank the following for their contributions to this work:

- Our advisors for strategic guidance
- Early customers for feedback and data access
- The open-source community for foundational tools
- The research community whose work forms the basis of this report

15.3 Funding

This work was supported by [funding sources to be added].

15.4 Competing Interests

The authors are employees or founders of AtlasPro AI, which is developing commercial products based on the research described in this report.

16 Extended Industry Analysis

This appendix provides comprehensive analysis of the industries AtlasPro AI targets.

16.1 Telecommunications Industry Overview

16.1.1 Market Structure

The telecommunications industry is undergoing a fundamental transformation driven by:

- **Fiber Expansion:** The shift from copper to fiber-to-the-home (FTTH) is accelerating. Global FTTH connections are projected to reach 1.5 billion by 2028, up from 900 million in 2023.
- **5G Deployment:** 5G networks require dense fiber backhaul, driving infrastructure investment.
- **Rural Broadband:** Government programs (BEAD, RDOF) are funding rural fiber deployment.
- **Consolidation:** The industry is consolidating, with larger players acquiring smaller ISPs.

16.1.2 Key Players

16.1.3 Pain Points

Telecommunications companies face several challenges that AtlasPro AI addresses:

1. **Network Complexity:** Modern networks include millions of components across diverse geographies.
2. **Aging Infrastructure:** Much of the existing infrastructure is decades old and poorly documented.

Table 17: Major Telecommunications Players

Company	Type	Fiber Miles	Customers
AT&T	Incumbent	2.5M+	15M+
Verizon	Incumbent	1.5M+	10M+
Lumen	Enterprise	450K+	Enterprise
Frontier	Regional	500K+	3M+
Ziply Fiber	Regional	100K+	500K+

3. **Skilled Labor Shortage:** Experienced network engineers are retiring faster than new ones are trained.

4. **Regulatory Pressure:** Governments are imposing stricter reliability and coverage requirements.

5. **Cost Pressure:** Competition is driving down prices while infrastructure costs rise.

16.2 Electric Utility Industry Overview

16.2.1 Market Structure

The electric utility industry is transforming due to:

- **Decarbonization:** Utilities are transitioning from fossil fuels to renewable energy.
- **Distributed Energy:** Rooftop solar and battery storage are changing grid dynamics.
- **Electrification:** Electric vehicles and heat pumps are increasing demand.
- **Grid Modernization:** Smart grid investments are enabling new capabilities.

16.2.2 Key Players

Table 18: Major Electric Utilities

Company	Type	Customers	Revenue
Duke Energy	IOU	8.2M	$29B
Southern Company	IOU	9M	$29B
Dominion Energy	IOU	7M	$17B
PG&E	IOU	5.5M	$24B
Xcel Energy	IOU	3.7M	$15B

16.2.3 Pain Points

Electric utilities face challenges that AtlasPro AI addresses:

1. **Grid Reliability:** Climate change is increasing extreme weather events that damage infrastructure.

2. **Renewable Integration:** Variable renewable generation creates grid stability challenges.
3. **Asset Management:** Utilities manage millions of assets with limited visibility into condition.
4. **Workforce Transition:** Experienced workers are retiring, taking institutional knowledge with them.
5. **Regulatory Compliance:** Utilities must meet strict reliability and safety standards.

16.3 Smart Cities and Urban Computing

16.3.1 Market Overview

Smart city investments are growing rapidly:

- Global smart city market: $820B by 2030 (from $410B in 2023)
- Key segments: Transportation, energy, water, public safety, governance
- Leading cities: Singapore, Seoul, Barcelona, Dubai, Copenhagen

16.3.2 Use Cases

1. **Traffic Management:** Real-time signal optimization, congestion prediction
2. **Public Transit:** Route optimization, demand forecasting, maintenance prediction
3. **Energy Management:** Building energy optimization, grid demand response
4. **Water Management:** Leak detection, demand forecasting, quality monitoring
5. **Public Safety:** Crime prediction, emergency response optimization

17 Detailed Technology Stack

This appendix provides detailed specifications for AtlasPro AI's technology stack.

17.1 Infrastructure Layer

17.1.1 Cloud Platform

- **Primary:** AWS (most customers are AWS-based)
- **Secondary:** GCP, Azure (for customer requirements)
- **Multi-cloud:** Kubernetes enables portability

17.1.2 Compute

- **Training:** NVIDIA A100/H100 GPUs for model training
- **Inference:** NVIDIA T4/L4 GPUs for production inference
- **CPU:** AMD EPYC for data processing workloads

17.1.3 Storage

- **Object Storage:** S3 for models, datasets, artifacts
- **Block Storage:** EBS for database volumes
- **File Storage:** EFS for shared model weights

17.2 Data Layer

17.2.1 Graph Database

- **Primary:** Neo4j Enterprise
- **Alternative:** Amazon Neptune, TigerGraph
- **Scale:** Billions of nodes and edges
- **Features:** ACID transactions, graph algorithms, full-text search

17.2.2 Time-Series Database

- **Primary:** InfluxDB or TimescaleDB
- **Scale:** Millions of metrics per second
- **Retention:** 5+ years of historical data
- **Features:** Downsampling, continuous queries, alerting

17.2.3 Vector Database

- **Primary:** Pinecone or Weaviate
- **Scale:** Billions of vectors
- **Features:** Hybrid search, filtering, metadata

17.3 ML Platform

17.3.1 Training Infrastructure

- **Orchestration:** Kubeflow or SageMaker
- **Experiment Tracking:** MLflow or Weights & Biases
- **Feature Store:** Feast or Tecton
- **Data Versioning:** DVC or LakeFS

17.3.2 Model Serving

- **Inference Server:** NVIDIA Triton
- **Model Registry:** MLflow Model Registry
- **A/B Testing:** Custom implementation
- **Monitoring:** Prometheus + custom metrics

17.4 Application Layer

17.4.1 Backend

- **Language:** Python 3.11+
- **Framework:** FastAPI
- **Task Queue:** Celery with Redis
- **Caching:** Redis

17.4.2 Frontend

- **Framework:** React with TypeScript
- **State Management:** Redux Toolkit
- **Mapping:** Mapbox GL JS
- **Visualization:** D3.js, Plotly

18 Comprehensive Benchmark Results

This appendix provides detailed benchmark results from our preliminary experiments.

18.1 GNN Model Benchmarks

18.1.1 Node Classification on Network Graphs

We evaluated GNN architectures on a synthetic network graph classification task:

Table 19: Node Classification Results (Synthetic Network Graph)

Model	Accuracy	F1 Score	Training Time
MLP (no graph)	72.3%	0.68	5 min
GCN	81.2%	0.78	15 min
GAT	83.5%	0.81	25 min
GraphSAGE	82.1%	0.79	20 min
AtlasPro GNN	86.7%	0.84	30 min

Key Findings:

- Graph structure provides significant improvement over MLP baseline (+14%)
- Attention mechanisms (GAT) outperform spectral methods (GCN)
- Our spatio-temporal architecture provides additional gains (+3%)

18.1.2 Link Prediction

We evaluated link prediction for network connectivity:

Table 20: Link Prediction Results

Model	AUC-ROC	AP
Node2Vec	0.82	0.79
GCN	0.87	0.84
GAT	0.89	0.86
AtlasPro GNN	0.92	0.89

18.2 Agent Benchmarks

18.2.1 Tool Use Accuracy

We evaluated agent accuracy on network analysis tasks:

Table 21: Agent Tool Use Accuracy

Task Type	GPT-4 Baseline	AtlasPro Agent
Information Retrieval	85%	95%
Single-step Analysis	72%	88%
Multi-step Analysis	58%	78%
Planning	45%	72%

Key Findings:

- Domain-specific tools significantly improve accuracy
- Multi-step tasks show largest improvement (+20%)
- Planning tasks benefit most from GNN integration

19 Regulatory and Compliance Considerations

This appendix discusses regulatory considerations for deploying AI in critical infrastructure.

19.1 Telecommunications Regulations

19.1.1 FCC Requirements

- **Network Reliability:** FCC requires 99.999% uptime for critical services
- **Outage Reporting:** Major outages must be reported within 120 minutes
- **CPNI:** Customer Proprietary Network Information must be protected
- **E911:** Emergency services must be maintained during outages

19.1.2 State PUC Requirements

- Service quality standards vary by state
- Reporting requirements for service metrics
- Rate case proceedings may require network data

19.2 Utility Regulations

19.2.1 NERC Standards

- **CIP Standards:** Critical Infrastructure Protection requirements
- **Reliability Standards:** Transmission planning and operations
- **Cyber Security:** Protection of bulk electric system cyber assets

19.2.2 State PUC Requirements

- Integrated Resource Planning (IRP) requirements
- Reliability standards and reporting
- Rate recovery for technology investments

19.3 AI-Specific Regulations

19.3.1 EU AI Act

- Critical infrastructure AI may be classified as high-risk
- Requirements for transparency, human oversight, accuracy
- Documentation and conformity assessment requirements

19.3.2 US AI Executive Order

- Safety and security requirements for AI in critical infrastructure
- Reporting requirements for large AI models
- Guidance on responsible AI deployment

19.4 AtlasPro AI Compliance Approach

1. **Human-in-the-Loop:** Critical decisions require human approval
2. **Audit Trail:** All AI recommendations are logged with explanations
3. **Model Documentation:** Comprehensive documentation of model capabilities and limitations
4. **Testing:** Rigorous testing before deployment in production
5. **Monitoring:** Continuous monitoring of model performance and drift

20 Intellectual Property Strategy

This appendix outlines AtlasPro AI's intellectual property strategy.

20.1 Patent Strategy

20.1.1 Core Innovations

We are pursuing patent protection for:

1. **Spatio-Temporal GNN Architecture:** Novel architecture for network failure prediction
2. **MCP Tool Orchestration:** Methods for coordinating AI agent tools
3. **Graph-Based Network Simulation:** World model for network planning
4. **Hierarchical Memory System:** Memory architecture for long-horizon spatial tasks

20.1.2 Filing Timeline

- Q1 2026: Provisional applications for core innovations
- Q1 2027: PCT applications for international protection
- Q2 2027: National phase entries in key markets

20.2 Trade Secrets

We protect the following as trade secrets:

- Training data preprocessing pipelines
- Model hyperparameters and training recipes
- Customer-specific model adaptations
- Benchmark datasets and evaluation protocols

20.3 Open Source Strategy

We contribute to open source to:

- Build community and attract talent
- Establish thought leadership
- Benefit from community contributions
- Reduce maintenance burden for non-core components

Open Source Contributions:

- Benchmark datasets (anonymized)
- Evaluation tools and metrics
- Reference implementations of baseline methods
- Documentation and tutorials

21 Team and Organization

This appendix provides information about the AtlasPro AI team.

21.1 Founding Team

Gloria Felicia, CEO & Co-Founder

- Background: [To be added]
- Expertise: AI research, product strategy
- Role: Overall strategy, research direction, fundraising

Nolan Bryant, CTO & Co-Founder

- Background: [To be added]
- Expertise: ML systems, infrastructure
- Role: Technical architecture, engineering leadership

Handi Putra, Chief Scientist & Co-Founder

- Background: [To be added]
- Expertise: GNNs, world models
- Role: Research leadership, model development

21.2 Advisory Board

[To be added]

21.3 Hiring Plan

Table 22: Hiring Plan (2026-2027)

Role	Timeline	Count
ML Engineer	Q1 2026	2
Backend Engineer	Q1 2026	2
Frontend Engineer	Q2 2026	1
Data Engineer	Q2 2026	1
Sales Engineer	Q3 2026	2
Customer Success	Q4 2026	2

22 Financial Projections

This appendix provides financial projections for AtlasPro AI.

22.1 Revenue Model

Pricing Structure:

- **Platform Fee:** $50K-$500K/year based on network size
- **Usage Fee:** $0.01-$0.10 per API call
- **Professional Services:** $200-$400/hour for implementation

22.2 Five-Year Projections

Table 23: Financial Projections ($M)

Metric	2026	2027	2028	2029	2030
ARR	0.5	2.5	8.0	20.0	45.0
Customers	3	12	35	80	150
Employees	8	20	45	80	120
Gross Margin	60%	70%	75%	78%	80%

22.3 Funding Requirements

- **Seed Round (2025):** $2M for MVP development and first customers
- **Series A (2026):** $10M for product expansion and go-to-market
- **Series B (2028):** $30M for scale and market expansion

22.4 Use of Funds

Seed Round Allocation:

- Engineering (60%): Core platform development
- Research (20%): Model development and benchmarking
- Operations (20%): Infrastructure, legal, admin

23 Detailed Comparison with Existing Solutions

This appendix provides detailed feature-by-feature comparisons with existing solutions.

23.1 GIS Platform Comparison

Table 24: GIS Platform Feature Comparison

Feature	Esri ArcGIS	QGIS	AtlasPro AI
Natural Language Interface	Limited	No	Yes
AI-Powered Analysis	Add-on	Plugin	Native
Predictive Analytics	Limited	No	Yes
Graph-Based Reasoning	No	No	Yes
Real-Time Processing	Limited	Limited	Yes
Agent Automation	No	No	Yes
World Model Simulation	No	No	Yes
Custom Model Training	Limited	No	Yes
API-First Design	Yes	Limited	Yes
Cloud-Native	Partial	No	Yes

23.2 Network Planning Tool Comparison

Table 25: Network Planning Tool Comparison

Feature	Comsof Fiber	3-GIS	AtlasPro AI
Automated Design	Yes	Limited	Yes
ML-Based Optimization	No	No	Yes
Failure Prediction	No	No	Yes
Demand Forecasting	Limited	No	Yes
Natural Language Queries	No	No	Yes
Learning from Data	No	No	Yes
Real-Time Updates	Limited	Yes	Yes
Multi-Network Support	Fiber only	Fiber only	Multi-utility

23.3 AI Platform Comparison

Table 26: AI Platform Comparison

Feature	Palantir	C3.ai	AtlasPro AI
Spatial-Native	No	No	Yes
GNN Support	Limited	Limited	Native

Feature	Palantir	C3.ai	AtlasPro AI
Network Domain Focus	No	Partial	Yes
Agentic AI	Limited	Limited	Yes
World Models	No	No	Yes
MCP Integration	No	No	Yes
Pricing	Enterprise	Enterprise	SMB-friendly
Implementation Time	Months	Months	Weeks

24 Extended Use Case Library

This appendix provides an extended library of use cases for AtlasPro AI.

24.1 Telecommunications Use Cases

24.1.1 Use Case T1: Proactive Maintenance Scheduling

Problem: Maintenance crews are dispatched reactively, leading to inefficient routing and overtime costs.

Solution: AtlasPro AI predicts failures 90 days in advance, enabling proactive scheduling that optimizes crew routing and reduces emergency dispatches.

Expected Impact:

- 40% reduction in emergency dispatches
- 25% improvement in crew utilization
- 30% reduction in overtime costs

24.1.2 Use Case T2: Capacity Planning

Problem: Network planners struggle to forecast demand growth and identify capacity bottlenecks before they cause service degradation.

Solution: AtlasPro AI uses spatio-temporal models to forecast demand growth and identify segments that will exceed capacity thresholds.

Expected Impact:

- 6-month advance warning of capacity constraints
- 20% reduction in emergency capacity upgrades
- Improved customer satisfaction through proactive upgrades

24.1.3 Use Case T3: Fiber Route Optimization

Problem: Designing optimal fiber routes is time-consuming and requires expert knowledge of local conditions.

Solution: AtlasPro AI generates optimal routes considering cost, reliability, and constructability, learning from historical construction data.

Expected Impact:

- 15% reduction in construction costs
- 80% reduction in design time
- Improved route quality through data-driven optimization

24.1.4 Use Case T4: Outage Root Cause Analysis

Problem: When outages occur, identifying the root cause requires manual investigation that delays restoration.

Solution: AtlasPro AI correlates network topology, telemetry, and environmental data to automatically identify probable root causes.

Expected Impact:

- 50% reduction in mean time to identify root cause
- 30% reduction in mean time to restore
- Improved first-time fix rate

24.2 Electric Utility Use Cases

24.2.1 Use Case E1: Vegetation Management

Problem: Vegetation contact is a leading cause of outages, but inspection and trimming are expensive.

Solution: AtlasPro AI analyzes satellite imagery, LiDAR, and historical outage data to prioritize vegetation management.

Expected Impact:

- 30% reduction in vegetation-related outages
- 20% reduction in vegetation management costs
- Improved targeting of high-risk areas

24.2.2 Use Case E2: Storm Damage Prediction

Problem: Severe weather causes widespread outages, but utilities struggle to pre-position crews effectively.

Solution: AtlasPro AI combines weather forecasts with network vulnerability models to predict damage locations.

Expected Impact:

- 40% improvement in crew pre-positioning accuracy
- 25% reduction in restoration time
- Improved customer communication through accurate ETRs

24.2.3 Use Case E3: DER Integration Planning

Problem: Distributed energy resources (solar, batteries) are being added rapidly, creating grid stability challenges.

Solution: AtlasPro AI models the impact of DER additions on grid stability and identifies necessary upgrades.

Expected Impact:

- Faster interconnection study completion
- Reduced hosting capacity violations
- Optimized upgrade investments

24.3 Smart City Use Cases

24.3.1 Use Case S1: Traffic Signal Optimization

Problem: Traffic signals are timed based on historical patterns, not real-time conditions.

Solution: AtlasPro AI uses GNN-based traffic prediction to optimize signal timing in real-time.

Expected Impact:

- 15% reduction in average travel time
- 10% reduction in emissions
- Improved emergency vehicle response times

24.3.2 Use Case S2: Public Transit Optimization

Problem: Transit agencies struggle to match service to demand, leading to overcrowding and empty buses.

Solution: AtlasPro AI forecasts ridership demand and optimizes routes and schedules.

Expected Impact:

- 20% improvement in service efficiency
- 15% increase in ridership
- Reduced operating costs

25 Detailed Safety Analysis

This appendix provides detailed analysis of safety considerations for AtlasPro AI.

25.1 Potential Risks

25.1.1 Risk Category 1: Model Errors

Description: AI models may make incorrect predictions that lead to poor decisions.

Examples:

- False negative: Failing to predict a failure that occurs

- False positive: Predicting a failure that doesn't occur
- Incorrect root cause: Misidentifying the cause of an outage

Mitigations:

- Uncertainty quantification: Flag low-confidence predictions
- Human-in-the-loop: Require human approval for critical decisions
- Continuous monitoring: Track model performance and retrain as needed

25.1.2 Risk Category 2: Adversarial Attacks

Description: Malicious actors may attempt to manipulate AI systems.

Examples:

- Data poisoning: Injecting false data to corrupt models
- Prompt injection: Manipulating agent behavior through crafted inputs
- Model extraction: Stealing proprietary models through API queries

Mitigations:

- Input validation: Sanitize all inputs before processing
- Access control: Limit API access to authorized users
- Anomaly detection: Monitor for unusual query patterns

25.1.3 Risk Category 3: Unintended Consequences

Description: AI recommendations may have unintended negative effects.

Examples:

- Optimization gaming: System exploits loopholes in objectives
- Cascading failures: Recommendation causes downstream problems
- Bias amplification: System perpetuates or amplifies existing biases

Mitigations:

- Simulation: Test recommendations in world model before execution
- Gradual rollout: Deploy changes incrementally with monitoring
- Bias auditing: Regular audits for fairness and bias

25.2 Safety Framework

AtlasPro AI implements a comprehensive safety framework:

Level 1: Input Validation

- All inputs are validated against expected schemas
- Anomalous inputs are flagged for review
- Rate limiting prevents abuse

Level 2: Model Guardrails

- Outputs are validated against physical constraints
- Uncertainty is quantified and communicated
- Fallback to conservative defaults when uncertain

Level 3: Human Oversight

- Critical decisions require human approval
- All recommendations include explanations
- Audit trail enables post-hoc review

Level 4: Continuous Monitoring

- Model performance tracked in real-time
- Drift detection triggers retraining
- Incident response procedures documented

26 Comprehensive Reference Architecture

This appendix provides a comprehensive reference architecture for AtlasPro AI deployments.

26.1 Logical Architecture

26.1.1 Data Ingestion Layer

Components:

- **GIS Connector:** Imports network topology from Esri, Smallworld, etc.
- **SCADA Connector:** Ingests real-time telemetry from SCADA systems
- **Weather Connector:** Fetches weather data from NOAA, commercial providers
- **Work Order Connector:** Imports maintenance records from work management systems

Data Flow:

1. Connectors poll source systems on configurable schedules
2. Raw data is validated and transformed
3. Transformed data is loaded into appropriate databases
4. Change events trigger model updates as needed

26.1.2 Storage Layer

Graph Database:

- Stores network topology as nodes and edges
- Supports complex graph queries (shortest path, centrality, etc.)
- Maintains historical versions for temporal analysis

Time-Series Database:

- Stores telemetry data (utilization, latency, etc.)
- Supports downsampling for long-term storage
- Enables fast range queries for analysis

Vector Database:

- Stores embeddings for semantic search
- Enables RAG for agent knowledge retrieval
- Supports hybrid search (vector + metadata)

26.1.3 Model Layer

GNN Models:

- Failure prediction model
- Capacity forecasting model
- Anomaly detection model
- Network embedding model

LLM Integration:

- Query understanding and intent classification
- Response generation and explanation
- Tool orchestration and planning

World Model:

- Network state simulation
- What-if analysis
- Planning and optimization

26.1.4 Agent Layer

MCP Server:

- Exposes tools for LLM agent use
- Handles authentication and authorization
- Logs all tool invocations for audit

Agent Orchestrator:

- Manages agent state and memory
- Implements ReAct-style reasoning loop
- Handles multi-step task decomposition

Safety Guardrails:

- Input validation and sanitization
- Output validation against constraints
- Human-in-the-loop for critical decisions

26.1.5 Application Layer

Web Application:

- Interactive map-based interface
- Natural language query interface
- Dashboard and reporting

API:

- RESTful API for programmatic access
- GraphQL for flexible queries
- Webhook support for integrations

26.2 Deployment Patterns

26.2.1 Pattern 1: Cloud-Native SaaS

Description: Fully managed deployment in AtlasPro AI's cloud.

Pros:

- Fastest time to value
- No infrastructure management
- Automatic updates and scaling

Cons:

- Data leaves customer environment
- Less customization flexibility
- Dependent on internet connectivity

26.2.2 Pattern 2: Customer VPC Deployment

Description: Deployed in customer's cloud account (AWS, GCP, Azure).

Pros:

- Data stays in customer environment
- Integrates with existing cloud infrastructure
- Customer controls security and compliance

Cons:

- Requires cloud expertise
- Customer responsible for infrastructure
- More complex updates

26.2.3 Pattern 3: On-Premises Deployment

Description: Deployed in customer's data center.

Pros:

- Maximum data control
- No cloud dependency
- Meets strict compliance requirements

Cons:

- Highest implementation complexity
- Customer responsible for all infrastructure
- Limited scalability

27 Conclusion and Call to Action

This technical report has presented AtlasPro AI's comprehensive approach to building autonomous spatial intelligence systems for critical infrastructure. We have demonstrated:

1. A clear market opportunity at the intersection of agentic AI and network intelligence
2. A differentiated technical approach based on GNNs, world models, and MCP integration
3. A systematic analysis of failure modes and mitigation strategies
4. A practical roadmap for building production-grade systems
5. Detailed specifications for implementation and deployment

We believe that spatial intelligence represents the next frontier for AI systems. The ability to perceive, reason about, and act within physical environments is essential for AI to have meaningful impact in the real world.

AtlasPro AI is committed to advancing this frontier, with an initial focus on the critical infrastructure sectors where our team has deep expertise and where the need for intelligent automation is most acute.

We invite collaboration from:

- **Researchers:** To advance the leading in spatial AI
- **Infrastructure Operators:** To validate our approach with real-world data
- **Investors:** To support our mission of building autonomous spatial intelligence
- **Talent:** To join our team and help build the future of infrastructure AI

For more information, please contact us at `research@atlaspro.ai`.

References

Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gober, Karol Gopalakrishnan, et al. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*, 2022.

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: A visual language model for few-shot learning. *Advances in Neural Information Processing Systems*, 35:23716–23736, 2022.

Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sunderhauf, Ian Reid, Stephen Gould, and Anton van den Hengel. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3674–3683, 2018.

Anthropic. Claude 3 model card. *Anthropic Technical Report*, 2024.

Iro Armeni, Zhi-Yang He, JunYoung Gwak, Amir R Zamir, Martin Fischer, Jitendra Malik, and Silvio Savarese. 3d scene graph: A structure for unified semantics, 3d space, and camera. *arXiv preprint arXiv:1910.02527*, 2019.

Favyen Bastani et al. Satlaspretrain: A large-scale dataset for remote sensing image understanding. In *arXiv preprint arXiv:2211.15660*, 2023.

Dhruv Batra, Aaron Gokaslan, Aniruddha Kembhavi, Oleksandr Maksymets, Roozbeh Mottaghi, Manolis Savva, Alexander Toshev, and Erik Wijmans. Objectnav revisited: On evaluation of embodied agents navigating to objects. In *arXiv preprint arXiv:2006.13171*, 2020.

Kevin Black, Noah Brown, Danny Driess, et al. pi0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

Anthony Brohan, Noah Brown, Justice Carbajal, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.

Tim Brooks et al. Video generation models as world simulators. *OpenAI Technical Report*, 2024.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33:1877–1901, 2020.

Jake Bruce, Michael Dennis, Ashley Edwards, et al. Genie: Generative interactive environments. In *ICML*, 2024.

Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, et al. nuscenes: A multimodal dataset for autonomous driving. In *CVPR*, 2020.

Devendra Singh Chaplot, Dhiraj Gandhi, Saurabh Gupta, Abhinav Gupta, and Ruslan Salakhutdinov. Learning to explore using active neural slam. In *ICLR*, 2020.

Boyuan Chen, Zhuo Xu, Sean Kirmani, Brian Ichter, Danny Driess, Pete Florence, Dorsa Sadigh, Leonidas Guibas, and Fei Xia. Spatialvlm: Endowing vision-language models with spatial reasoning capabilities. *arXiv preprint arXiv:2401.12168*, 2024a.

Zhiqiang Chen et al. Dust: A framework for data-driven urban simulation and traffic. *arXiv preprint*, 2024b.

Yezhen Cong, Samar Khanna, Chenlin Meng, Patrick Liu, Erik Rozi, Yutong He, Marshall Burke, David Lobell, and Stefano Ermon. Satmae: Pre-training transformers for temporal and multispectral satellite imagery. *Advances in Neural Information Processing Systems*, 35:197–211, 2022.

Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Niessner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5828–5839, 2017.

Caelan Reed Garrett et al. Integrated task and motion planning. *Annual Review of Control*, 2021.

Justin Gilmer et al. Neural message passing for quantum chemistry. *International Conference on Machine Learning*, 2017.

Ankit Goyal, Jie Xu, Yijie Guo, Valts Blukis, Yu-Wei Chao, and Dieter Fox. Rvt-2: Learning precise manipulation from few demonstrations. In *RSS*, 2024.

Ankit Goyal et al. Rvt: Robotic view transformer for 3d object manipulation. In *CoRL*, 2023.

Kelvin Guu et al. Realm: Retrieval-augmented language model pre-training. *ICML*, 2020.

David Ha and Jurgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2018.

Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models. In *ICLR*, 2021.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023.

Danijar Hafner et al. Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603*, 2019.

William L Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in Neural Information Processing Systems*, 30, 2017.

Shibo Hao, Yi Gu, Haodi Ma, Joshua Jiahua Hong, Zhen Wang, Daisy Zhe Wang, and Zhiting Hu. Reasoning with language model is planning with world model. *arXiv preprint arXiv:2305.14992*, 2023.

Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Ceyao Zhang, Jinlin Wang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, et al. Metagpt: Meta programming for a multi-agent collaborative framework. *arXiv preprint arXiv:2308.00352*, 2023.

Yicong Hong et al. Recurrent models of visual attention for vision-and-language navigation. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

Anthony Hu et al. Gaia-1: A generative world model for autonomous driving. *arXiv preprint arXiv:2309.17080*, 2023.

Chenguang Huang, Oier Mees, Andy Zeng, and Wolfram Burgard. Visual language maps for robot navigation. *arXiv preprint arXiv:2210.05714*, 2023a.

Wenlong Huang, Chen Wang, Ruohan Zhang, Yunzhu Li, Jiajun Wu, and Li Fei-Fei. Voxposer: Composable 3d value maps for robotic manipulation with language models. *arXiv preprint arXiv:2307.05973*, 2023b.

Johannes Jakubik, Sujit Roy, C E Phillips, Paolo Fraccaro, Denys Godwin, Bianca Zadrozny, Daniela Szwarcman, Carlos Gomes, Gabby Musber, Daiki Oliveira, et al. Prithvi: A foundation model for earth observation. *arXiv preprint arXiv:2310.18660*, 2024.

Stephen James, Zicong Ma, David Rovick Arrojo, and Andrew J Davison. Rlbench: The robot learning benchmark. *IEEE Robotics and Automation Letters*, 2020.

Stephen James, Kentaro Wada, Tristan Laidlow, and Andrew J Davison. Coarse-to-fine q-attention: Efficient learning for visual robotic manipulation via discretisation. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

Guangyin Jin, Yuxuan Liang, Yuchen Fang, Zezhi Huang, Junbo Zhang, and Yu Zheng. Spatio-temporal graph neural networks for urban computing: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 2023.

Subbarao Kambhampati, Karthik Valmeekam, Lin Guan, Mudit Verma, Kaya Stechly, Siddhant Bhambri, Lucas Saldanha, and Anil Murthy. Llms can't plan, but can help planning in llm-modulo frameworks. *arXiv preprint arXiv:2402.01817*, 2024.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2017.

Barbara Kitchenham. Procedures for performing systematic reviews. *Keele University Technical Report*, 2004.

Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. In *International Journal of Computer Vision*, volume 123, pages 32–73, 2017.

Alexander Ku, Peter Anderson, Roma Patel, Eugene Ie, and Jason Baldridge. Room-across-room: Multilingual vision-and-language navigation with dense spatiotemporal grounding. In *EMNLP*, 2020.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Kuttler, Mike Lewis, Wen-tau Yih, Tim Rocktaschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474, 2020.

Guohao Li, Hasan Abed Al Kader Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. Camel: Communicative agents for mind exploration of large language model society. *arXiv preprint arXiv:2303.17760*, 2023.

Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In *International Conference on Learning Representations*, 2018.

Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. Code as policies: Language model programs for embodied control. *arXiv preprint arXiv:2209.07753*, 2023.

Bo Liu, Yuqian Jiang, Xiaohan Zhang, Qiang Liu, Shiqi Zhang, Joydeep Biber, Jian Zhao, and Peter Stone. Llm+p: Empowering large language models with optimal planning proficiency. *arXiv preprint arXiv:2304.11477*, 2023a.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *arXiv preprint arXiv:2304.08485*, 2023b.

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, et al. Agentbench: Evaluating llms as agents. *arXiv preprint arXiv:2308.03688*, 2023c.

Gengchen Mai, Weiming Huang, Jin Sun, Suhang Song, Deepak Mishra, Ninghao Liu, Song Gao, Tianming Liu, Gao Cong, Yingjie Hu, et al. Opportunities and challenges of foundation models for geospatial artificial intelligence. *arXiv preprint arXiv:2304.06798*, 2023.

Arjun Majumdar, Ayush Shrivastava, Stefan Lee, Peter Anderson, Devi Parikh, and Dhruv Batra. Improving vision-and-language navigation with image-text pairs from the web. *European Conference on Computer Vision*, 2020.

David Marr. *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information.* MIT Press, 1982.

Oier Mees, Lukas Hermann, Erick Rosete-Beas, and Wolfram Burgard. Calvin: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks. In *IEEE RA-L*, 2022.

OpenAI. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G Patil, Ion Stoica, and Joseph E Gonzalez. Memgpt: Towards llms as operating systems. *arXiv preprint arXiv:2310.08560*, 2023.

Shishir G Patil, Tianjun Zhang, Xin Wang, and Joseph E Gonzalez. Gorilla: Large language model connected with massive apis. *arXiv preprint arXiv:2305.15334*, 2023.

Kai Petersen et al. Systematic mapping studies in software engineering. *EASE*, 2008.

Xavier Puig, Eric Undersander, Andrew Szot, Mikael Dallaire Cote, Dhruv Batra, and Roozbeh Mottaghi. Habitat 3.0: A co-habitat for humans, avatars and robots. In *International Conference on Learning Representations*, 2024.

Yuankai Qi, Qi Wu, Peter Anderson, Xin Wang, William Yang Wang, Chunhua Shen, and Anton van den Hengel. Reverie: Remote embodied visual referring expression in real indoor environments. In *CVPR*, 2020.

Yifeng Qiao et al. Hopt: Hierarchical object-centric planning for task-oriented manipulation. *arXiv preprint*, 2023.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, et al. Toolllm: Facilitating large language models to master 16000+ real-world apis. *arXiv preprint arXiv:2307.16789*, 2023.

Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, et al. Habitat: A platform for embodied ai research. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9339–9347, 2019.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessi, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. *arXiv preprint arXiv:2302.04761*, 2023.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *arXiv preprint arXiv:2303.11366*, 2023.

Mohit Shridhar, Lucas Manuelli, and Dieter Fox. Cliport: What and where pathways for robotic manipulation. In *CoRL*, 2022.

Mohit Shridhar et al. Perceiver-actor: A multi-task transformer for robotic manipulation. In *CoRL*, 2023.

Tom Silver et al. Generalized planning in pddl domains with pretrained large language models. *AAAI Conference on Artificial Intelligence*, 2024.

Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al. Scalability in perception for autonomous driving: Waymo open dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2446–2454, 2020.

Andrew Szot, Alexander Clegg, Eric Undersander, et al. Habitat 2.0: Training home assistants to rearrange their habitat. In *NeurIPS*, 2021.

Jiabin Tang, Yuhao Yang, Wei Wei, Lei Shi, Lixin Su, Suqi Cheng, Dawei Yin, and Chao Huang. Graphgpt: Graph instruction tuning for large language models. *arXiv preprint arXiv:2310.13023*, 2024.

Gemini Team and Google. Gemini: A family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

Gemini Team, Rohan Anil, Sebastian Borgeaud, et al. Gemini: A family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

Octo Model Team et al. Octo: An open-source generalist robot policy. *arXiv preprint arXiv:2405.12213*, 2024.

Hugo Touvron, Louis Martin, Kevin Stone, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Karthik Valmeekam, Matthew Marquez, Sarath Sreedharan, and Subbarao Kambhampati. On the planning abilities of large language models: A critical investigation. *Advances in Neural Information Processing Systems*, 36, 2023.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2018.

Costas Wang et al. Gnn-rag: Graph neural retrieval for large language model reasoning. *arXiv preprint arXiv:2405.20139*, 2024a.

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, et al. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 2024b.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. Emergent abilities of large language models. *Transactions on Machine Learning Research*, 2022.

Benjamin Wilson et al. Argoverse 2: Next generation datasets for self-driving perception and forecasting. *Advances in Neural Information Processing Systems*, 2023.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, et al. Autogen: Enabling next-gen llm applications via multi-agent conversation. *arXiv preprint arXiv:2308.08155*, 2023.

Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, and Chengqi Zhang. Graph wavenet for deep spatial-temporal graph modeling. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, pages 1907–1913, 2019.

Rui Yang, Hanyang Lin, Junyu Zhu, and Jingyi Huang. Embodiedbench: Comprehensive benchmarking multi-modal large language models for vision-driven embodied agents. *arXiv preprint arXiv:2502.09560*, 2025.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2023.

Sheng Yin, Xianghe Xiong, Wenhao Huang, et al. Safeagentbench: A benchmark for safe task planning of embodied llm agents. *arXiv preprint arXiv:2412.13178*, 2025.

Bing Yu, Haoteng Yin, and Zhanxing Zhu. Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*, pages 3634–3640, 2018.

Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, et al. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *CoRL*, 2020.

Andy Zeng, Pete Florence, Jonathan Tompson, Stefan Welker, Jonathan Chien, et al. Transporter networks: Rearranging the visual world for robotic manipulation. In *CoRL*, 2021.

Yu Zheng, Licia Capra, Ouri Wolfson, and Hai Yang. Urban computing: Concepts, methodologies, and applications. *ACM Transactions on Intelligent Systems and Technology*, 5(3):1–55, 2014.