Bifröst: Spatial Networking with Bigraphs

Anonymous Author(s)*

Abstract

Modern networked environments increasingly rely on spatial reasoning, but lack a coherent representation for coordinating physical space. Consequently, tasks such as enforcing spatial access policies remain fragile and manual. We first propose a unifying representation based on bigraphs, capturing spatial, social, and communication relationships within a single formalism, with user-facing tools to generate bigraphs from physical environments. Second, we present a hierarchical agent architecture for distributed spatial reasoning, with runtimes for agentic processes to interact the spatial representation, and a context-aware execution model that scopes reasoning to the smallest viable subspace. Together, these enable private, reliable, and low-latency spatial networking that can safely interact with agentic workflows.

1 The Spatial Disconnect

Let us imagine what a future with seamless use of spatial networks might look like. Consider an office meeting room equipped with an electronic display, a smart microphone, a local tablet, and the personal devices of the staff members present. As staff enter the meeting room, the display automatically configures to show a shared folder of relevant documents, and recording begins only once all participants are present. If an unauthorized person enters the room, the display *immediately* blanks within "the uncanny valley of human perception" [17], with no awkward or intrusive delays. The potentially sensitive data on who is present and the meeting transcription does not leave the physical space without the explicit consent of those involved. Today, achieving this requires a complex, manual setup: device pairing via disparate applications, room-specific rules, and redundant per-device configurations.

The problem is that today's digital infrastructure lacks a unified representation for entities in physical space, despite networked devices being omnipresent. Advances in mobile and wearable hardware—including smartphones and VR/AR headsets—have enabled modeling real-world environments [3]. A class of 'spatial devices' is becoming ubiquitous in our networks that derives its identity from its location; a networked speaker in room 1.01 *is* the room 1.01 speaker. However, platforms [9, 21] for managing these devices do not offer a programming model over physical spaces, and rely on centralized coordination nodes with implications for privacy, reliability, and latency. There has been work on *naming* devices by their physical location [6] but there is still no framework for expressing policies across space.

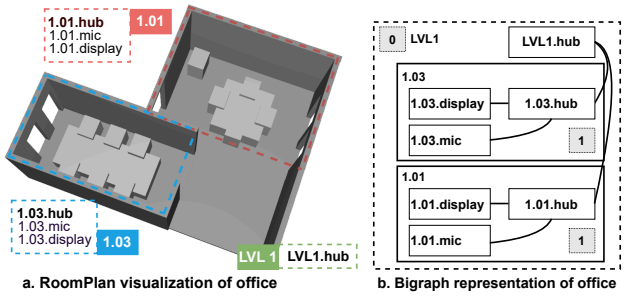

Figure 1: A 3D office map, generated on iPhone using Apple's RoomPlan, with its corresponding bigraph.

In this work, we propose bridging the virtual and physical worlds with *Bifröst*, a framework based on a representation of physical space with *bigraphs* [20], which capture the space and motion of communicating agents. A place graph generated from a 3D mobile scan (Fig. 1) is populated with access rights, device links, and behavioral policies to form a bigraph, where rules such as "enable file access and activate the local display when all authorized participants are present in the room" become both expressive and portable.

Autonomous agents can operate on this bigraph according to the "*principle of least context*": the system only escalates or shares physical data to the level necessary for the task at hand. Computational decisions move through nested levels of spatial agency, scaling context and compute only when needed. Such a framework is now feasible with low-power neural hardware [19], providing the computational headroom for efficient local reasoning; and breakthroughs in the capabilities of agents—in reasoning [31], planning [4], and tool use [25]—offer the foundation for agents to operate autonomously over complex, dynamic spaces.

Bifröst enables spatial networking that is:

- **Private**: Data remains local unless escalation is explicitly required, minimizing unnecessary exposure.
- **Reliable**: Distributed agents reduce dependence on a single node and functionality persists without cloud services.
- **Low-latency**: On-device reasoning supports sub-second responsiveness for immediate tasks.

In this paper, we describe how using bigraphs as a unified representation of the physical world enables reasoning spatial constraints (§2). Agents can interact with this representation (§3.1), applying localized reaction rules (§3.2), and escalating context (§3.3). We explore the implications (§4) and challenges (§5) of deploying Bifröst.

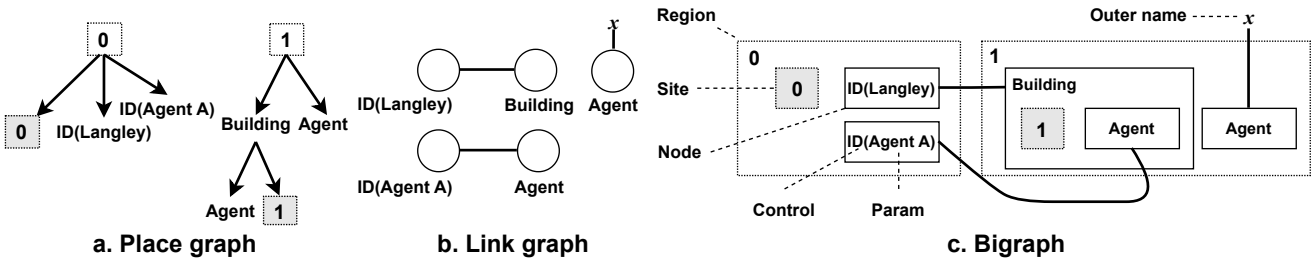

Figure 2: A *bigraph*, constructed from its underlying *place* and *link* graphs.

2 Connecting Space with Bigraphs

Consider a university lab with shared equipment such as a 3D printer, a smart whiteboard and a media console. Access policies depend on who is present, their role, the time of day, and collaborative context; e.g., the 3D printer should only operate when a trained technician is present, but students also working there can view or annotate the whiteboard feed. Specifying such nuanced, context-sensitive behaviors needs a formalism to represent physical space and containment, dynamic relationships, and evolving access intent. Current systems rely on ad-hoc rules tied to device IDs and fixed locations, or push complexity into fragile orchestration layers.

We propose using bigraphs [20] as a formalism that provides a concise, compositional representation of spatial nesting, containment, and connectivity. Formally, a bigraph consists of two orthogonal structures over the same set of typed entities. The *place* graph is a rooted forest representing a nested spatial hierarchy, with roots naturally modeling distinct adjacent *regions*. The *link* graph is a hypergraph connecting entity ports: links capture non-spatial relationships such as communication or data flow. For instance, a Wi-Fi network or a social link between nodes can be modelled as a hyperedge joining multiple ports.

Joining the link graph with the place graph enables the modeling of scenarios where actions or policies depend on both location and non-spatial connections. Bigraphs include closed links—complete hyperedges connecting nodes—alongside open links, which are incomplete hyperedges terminating in *outer names* (e.g., x in Fig. 2). Each node in a bigraph is associated with a *control*, defining its type and interface.

Bigraphs are also inherently dynamic: *reaction rules* specify how subgraphs update in response to events. For example, a rule can represent a person moving between connected places, automatically severing links when crossing a boundary. These rules and connectivity are inherently local: agents can make decisions based on their partial view of the graph, without requiring global knowledge, enabling distributed control and local reasoning.

Bigraphs are a particularly well-suited formalism for spatial coordination as they are:

- **Spatial**: Physical containment is explicitly and formally captured which allows policies and behaviors to respect spatial constraints, such as rules that apply only within a specific place.
- **Dynamic**: Bigraph reaction rules provide a mechanism to program dynamic changes in physical environments in real-time. Agents can efficiently update only the relevant portions of the bigraph they observe, and updates propagate via scoped synchronization mechanisms, each node maintaining a partial view of the global state and sharing updates with its neighbors as needed [18].
- **Interpretable**: Bigraphs support formal verification of spatial configurations such as checking policy consistency, constraint satisfiability, or detecting resource conflicts, enabling reliable, interpretable, and secure coordination.
- **Composable**: Policy reuse and reasoning across spatial and organizational domains is supported by bigraph composition.

The bigraph allows agents to act on a common but appropriately scoped spatial representation of the environment. For example, consider a local node updating its subgraph to reflect that a user is now present within its space. A reasoning agent, observing this change in its local view, can then access attributes of both the space and the user to mediate access control.

3 Spatially-Scoped Agents

The end users of these formalisms are increasingly AI-driven agents that are (hopefully) performing actions on behalf of humans. Although large language models (LLMs) excel at free-form text generation, ensuring that they produce structured, context-valid outputs is nontrivial [15]. For agents to interact effectively with real physical environments, and to avoid overwhelming LLM context windows, they require a queryable and up-to-date world representation. We use bigraphs to design an algorithm that balances how much context to provide an agent about the world.

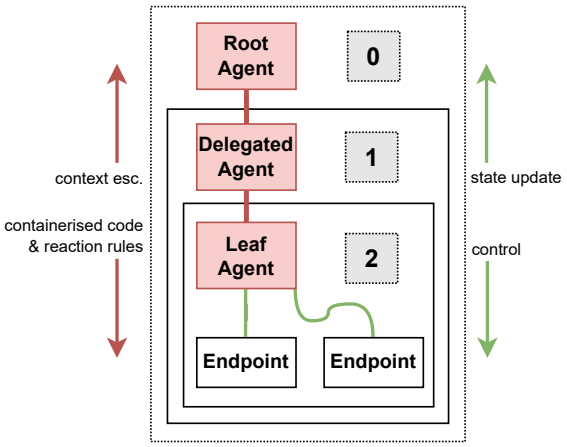

Figure 3: Our hierarchical agent architecture where agents coordinate via a shared *bigraph* 'world representation'.

Fig. 3 illustrates this "principle of least context". *Leaf* agents handle immediate task control whenever possible, using lightweight reaction rules or local models (e.g., for gesture recognition). Processing is escalated to higher-tier, but still local, *delegated* agents—operating within the scope of, say, a building—only when necessary for more context-aware inference, such as selecting which shared folder to display based on recognized participants, ongoing projects, and calendar events. Finally, larger *central* agents are invoked only for setup, complex reasoning, or policy updates. Policies encoded as reaction rules and code are pushed down the hierarchy to support private, reliable, low-latency operation. Agents interact with endpoints directly as a side-effect of reaction rules—for example, turning on a display—and endpoints can in turn signal agents to update the bigraph upon, say, detecting a person entering a room.

Using bigraphs as a unified representation of the world (§2), we define a framework for context-driven automation in spatial environments (§3.1), applying localized transformations (§3.2), with minimal context escalation (§3.3).

3.1 Contextual Reasoning over Bigraphs

While leaf agents handle immediate, reactive events, many desirable behaviors call for a richer understanding of context beyond simple triggers. However, invoking large parameter LLMs for every scenario where more than basic rule-matching is required is inefficient and latent, but also raises concerns around privacy and scalability. Instead, lightweight, localized reasoning can be applied.

In our framework, mid-tier agents (e.g., Qwen3-1.8B [5]) handle semantic and contextual reasoning over the bigraph layer. Their contextual reasoning is more powerful than simple reaction-rule-based logic but still operates over a localized graph view—that is, on abstracted data within a defined spatial scope such as a place, as represented by a relevant subgraph. For example, consider our coworking space scenario: in a meeting room, a delegated agent (running on a local machine in the *MeetingRoom* space) could automatically infer the shared folder to project by correlating recognized individuals (real-time data from face recognition nodes in the space) with ongoing projects and calendar events (contextual data linked to the user nodes in the bigraph layer). This decision is made locally, within the context of the room and its occupants, and without incurring the overhead or privacy risks associated with cloud processing.

LLM agents interact with bigraphs via a model context protocol [2] server that exposes the bigraph structure through a tooling interface. This allows agents to reason, update, or query the bigraph by invoking abstracted functions rather than directly manipulating graph data structures. Reaction rules can be formally verified for correctness or to ensure invariants hold prior to deployment and execution. Distributing the burden of reasoning across many localized agents avoids central bottlenecks, promotes privacy by design, and allows the system to scale more effectively to larger, more complex environments.

3.2 Localized Reaction Rules

To operationalize policies and automate behaviors responsively and reliably, our framework supports pushing executable logic, in the form of (user- or agent-generated) bigraphical reaction rules, to endpoints. This enables policies to be enforced locally and instantaneously, without requiring a centralized server to continuously poll states or orchestrate behavior changes. This unfolds into four stages:

- **Policy Definition**: High-level policies are defined from user preferences (e.g., "in my office, turn on focus lighting when I'm present and my calendar shows deep work"), building management directives ("during off-hours, dim lights in unoccupied common areas"), safety/privacy regulations ("pause all recording if an unconsenting individual enters a private meeting"), or generated by agents based on inferred user routines ("the user normally comes into work at 8am; heating should come on at 7am to ensure the office is at a comfortable temperature"). These policies can originate in natural language or a specific policy language.
- **Translation to Reaction Rules**: Abstract policies are compiled into formal bigraphical reaction rules. This could be performed by a higher-level agent, or by a compiler from policy descriptions to reaction rules.
- **Distribution to endpoints**: The reaction rules are pushed to the relevant endpoints that manage the entities involved

in the rule. This ensures that only the nodes that need to know about a particular policy, and have the local context to enforce it, receive the corresponding rule.
- **Local Execution**: Nodes perform incremental computation over their local bigraph state to match against their assigned reaction rules. If a match occurs, the rule is applied, updating the local bigraph and executing any associated side-effects. This ensures policies are enforced locally in real-time, without a central policy engine. This localization also allows policy enforcement to continue autonomously, even during connectivity failures to higher-level systems.

Recent work has shown that LLMs can synthesize executable Internet of Things (IoT) programs from natural language prompts [27], and can also assist in building, training, and deploying compact local models [29, 32]. These local models can serve as policy compilers, generating the necessary "glue code" and control logic tailored to the spatial and behavioral context encoded in the bigraph. The resulting code artifacts—along with any lightweight local models (e.g., gesture recognizers)—are packaged into container templates for deployment [17]. These are then propagated down the spatial hierarchy to the relevant nodes, enabling automatic provisioning or updates of runtime behavior. As user preferences or policies evolve, containers can be recompiled and re-deployed dynamically, allowing the system to adapt in real-time across the distributed environment.

```
react shutdown_nodes =
/x (MeetingRoom.(Users.() || Node{x} || rest))
--> MeetingRoom.(rest);

begin brs
init ...;
rules = [{shutdown_nodes}, {...}];
end
```

Listing 1: A BigraphER [26] reaction rule: all nodes in `MeetingRoom` are shut down when no users are present.

3.3 Context Escalation

In our framework, agents are distributed across spatial and semantic tiers – endpoints and local hubs to central organization-level servers and cloud-based agents—with our core design principle being that agency should always occur at the lowest viable tier. However, there are inevitably scenarios where an agent lacks sufficient context, authority, or capacity to act locally. In such cases, decision-making should *escalate* to a higher-tier agent.

3.3.1 From Leaf to Delegated Agents. At the lowest tier, local models—often running directly on nodes—are responsible for recognizing immediate patterns or events, such as motion, gestures, or faces. Decision logic is encoded in reaction rules, which specify local actions to recognized patterns. Escalation from this tier occurs under two conditions:

- **Rule-Driven**: Some rules contain escalation clauses. For example, "if more than one user is detected in the space, escalate with identifiers" This delegates higher-level decision-making (e.g., whether to initiate a meeting recording) to a higher-tier agent with access to broader context such as calendars, roles, or prior user/spatial activity.
- **Unknown or Ambiguous State**: If a detected state has no matching reaction (e.g., an unrecognized face or a novel combination of occupants), then the leaf agent cannot safely act autonomously and must escalate. This enables graceful handling of novel scenarios.

3.3.2 Between Delegated & Central Agents. Lower and mid-tier agents maintain state, can correlate data across nodes, and have access to broader reaction rule sets or symbolic models. However, they still operate over a constrained spatial scope (i.e., graph view). Escalation between agents (e.g., from place-level to organization- or cloud-level) occurs under two main conditions:

- **Self-Assessed Uncertainty**: The agent is explicitly unsure of the correct action. This may arise from lacking context, encountering unexpected input, needing multi-step reasoning beyond its capabilities, or other ambiguity. This uncertainty, however, is not inferred implicitly but instead surfaced using explicit self-reflection mechanisms (e.g., prompting the agent to evaluate its own confidence).
- **Policy Scope Violation**: The agent detects that the inferred context extends outside its authorized policy or spatial scope. This is managed with policy manifests, which detail the semantic and spatial scope of each agent's authority; if an inferred intent references resources, users, or dependencies beyond the agent's jurisdiction, a formal scope violation is raised and escalation is enforced. For example, if a place-level agent notices that an event involves users from different departments with separate policy domains, it should escalate to an organizational-level coordinator.

3.3.3 Scoped Escalation. To maintain privacy and responsiveness, escalation is never indiscriminate. Lower-tier agents communicate upward via bounded messages that respect abstraction boundaries, rather than streaming raw data. This is enforced via several complementary mechanisms:

- **Schema Contracts**: All escalation is functionally scoped, with agents communicating upwards via explicit escalation interface schemas that define the fields and data types permitted in upward messages. These schemas are enforced statically (via type-checking) and at runtime (via

schema validation); attempts to escalate data outside this schema are rejected.
- **Context Scoping**: Agents are provisioned with signed capability tokens that define their access rights and escalation privileges. These tokens are verified by higher-tier agents before escalation data is accepted, ensuring both policy compliance and provenance.
- **Auditability & Hash Tracing**: Each escalation's payload is hashed and logged alongside its originating node/agent ID. These logs support offline verification that only compliant messages are escalated, and can be sampled or attested for auditability.

This structured approach ensures escalation remains bounded, interpretable, and an enforceable process.

The asymmetric awareness across tiers is central to our framework's privacy goals. Lower-tier models possess raw data but have a narrow field of authority and policy scope. Higher-tier agents possess broader context and reasoning capacity, but with coarser access to raw data. Escalation allows coordination across these boundaries – but always under strict policy, schema, and trust constraints. By structuring escalation as a formally typed, policy-bounded process, we ensure agents escalate not only when necessary, but also only with the least context needed.

4 Implications

Bifröst embeds privacy by design: since the bigraph layer explicitly delimits which entities co-locate and which entities are allowed to interact, sensitive data and policies can be confined to local regions. Critically, data only crosses region boundaries when an explicit link is present, so any information flow is intentional and governed by reaction rules. This means our coordination model can enforce personal data stay within private subgraphs unless allowed, reducing inadvertent leaks. Accordingly, responsiveness is improved: since reasoning is local and often rule-based, agents avoid round-trip delays to a central controller. Instead, actions propagate along the bigraph only as needed, remaining agile to contextual changes.

The framework's reliability also benefits from its distributed and formally defined semantics. There is no single point of failure—places and agents operate autonomously—and the bigraph semantics permit exhaustive analysis of behaviors. In fact, by casting IoT coordination rules as bigraph reaction rules, we can use formal verification to prove properties like correctness or safety [1]. The framework is also highly compositional: new nodes or locations can be added by composing additional sub-bigraphs, and policies automatically merge with existing ones.

We now explore applications of this framework in three domains, spatial devices (§4.1), mobile devices (§4.2), and agent-driven automation (§4.3).

4.1 Spatial Devices

There's a class of stationary infrastructure that derives its identity from its location—such as displays, printers, speakers, and smart switches. The bigraph spatial representation enables the automatic generation of spatially-scoped names for these devices from the place graph. For example, a projector located in the building 1, floor 1, room A subgraph, can be named `projector.room-a.floor-1.building-1`, allowing agents and users to reference devices by their position in the containment hierarchy.

Prior work has explored the value of incorporating physical space as a first-class property in network architectures. Gibb et al. [6], for example, extend DNS to assign hierarchical location-based names, supporting spatial discovery and routing. This structured naming avoids reliance on static IPs or manually configured hostnames, and enables spatial queries to be resolved locally and efficiently. Spatial naming supports the transparent replacement of devices: when a device is swapped or upgraded, the new hardware automatically inherits the spatial name tied to its location, maintaining continuity of identity and service without manual reconfiguration. Our bigraph representation provides such names.

4.2 Mobile Devices

Mobile devices—such as health-monitoring wearables, AR headsets or emerging generalist robots—introduce unique spatial challenges. Unlike stationary devices, their connectivity, physical containment, and context shift constantly. Our framework supports spatially-aware mobile ad-hoc networks, via dynamic graph updates, scoping each agent's world representation to its current spatial region and linked neighbors. This dynamicity keeps reasoning and policy enforcement accurate and relevant as devices move—key for emerging mobile applications fusing on-device perception with cloud-assisted reasoning.

Nascent robotics pipelines [28], for example, exploit large vision models (LVMs) and vision–language models (VLMs) to infer a robot's location and task context. Yet, recent benchmarks [30] show multimodal LLMs (MLLMs) exhibit subhuman visual-spatial reasoning.Our framework augments such pipelines with formally grounded and locally enforced spatial reasoning, enabling mobile agents to reason about their environment via unified spatial representations while augmenting potentially unreliable MLLMs. Picture a mobile robot operating in a hospital or office; while MLLMs assist with general context understanding, the robot also queries its local spatial graph to obtain access policies and allowed

traversal paths—temporarily gaining access to a restricted corridor while delivering supplies but automatically losing it on exit, for example. Multiple robots can form temporary ad-hoc links to coordinate and share task assignments.

4.3 Agent-Driven Automation

Most smart home platforms rely on cloud-based voice assistants [7], which offer convenience but introduce latency and privacy concerns. Hub-based platforms [9, 21] can run discovery, automation and control locally, improving reliability and privacy, but still involve a single point of failure. They also often require considerable user configuration and technical proficiency to set up and maintain.

Both types of platform mostly rely on simple, rule-based automations and lack semantic understanding; they struggle to handle complex or ambiguous user requests that fall outside pre-programmed routines. Recent work has explored agent-oriented programming and coordination of smart environments as an alternative. GPIoT [27] employs a fine-tuned LLM (Llama2-13b) to generate IoT programs and their behaviors from high-level user requirements. Other multi-agent and semantic IoT frameworks aim to let users specify goals or context rules and have autonomous agents interpret and fulfil them. For instance, Sasha is an LLM-driven home assistant that can translate under-specified user requests (e.g., "make it cosy in here") into multi-step plans using available infrastructure and data [11]. Sage employs an LLM to orchestrate discrete smart home APIs and queries; it grounds the assistant's decisions in a dynamically structured prompt tree, achieving higher task success rates than prior rule-based or LLM-only baselines [23].

These agentic frameworks support more dynamic, adaptive behavior than rule-based platforms. However, most early works rely on fixed topologies and powerful cloud models, and raises new concerns around privacy and trustworthiness [13]. Moreover, such frameworks can be complex to integrate at scale, often lacking a clear coordination layer dealing with interactions among agents, instead relying on loosely coupled messaging or reasoning in text or code. Processing data locally, close to where it's generated—rather than streaming to the cloud—alleviates reliance on external servers, often improving latency and data privacy [10]. Lightweight ML architectures [14, 16, 24] and compression techniques [8, 12, 22], such as pruning and distillation, have allowed practical deployment of ML (e.g., for tasks like gesture recognition) on constrained hardware (e.g., MCUs). Moreover, approaches for adapting inference dynamically based on input complexity and workload, alongside the use of specialized hardware accelerators [19], have addressed real-time processing constraints.

5 Challenges

There are important challenges in deploying our design. The most obvious is an increase in modeling and configuration overhead compared to monolithic or centralized frameworks. There is considerably more up-front effort in setup, particularly in encoding spatial structure, policies, and coordination logic into bigraphical forms. While one can hand-design the bigraph for a given site, practical deployment at scale would benefit from automation, especially in dynamic environments like offices or homes.

However, automatic bigraph construction is nontrivial: building the graph layer from raw data or maps requires robust discovery methods. LiDAR-based indoor mapping tools (e.g., Apple's RoomPlan API [3]), RSSI fingerprinting, and even ML-based floor-plan annotations could be used to help build the underlying place graph. Fig. 1 details a 3D map of an office environment generated on an iPhone with RoomPlan, and includes the various IoT devices located in each of its places (e.g., meeting rooms 1.01 and 1.03). While generating the place graph alone is straightforward (e.g., RoomPlan's output contains a JSON of the map's hierarchical structure), augmenting this to include the various IoT devices is not; connectivity heuristics or proximity learning could infer the link structure.

We can infer social graphs overlaid on the bigraph by analysing co-occurrence data. Unlike a centralized system, with globally accessible logic and state, our framework relies on distributed management and orchestration of coordination logic (i.e., what each agent knows, what it is allowed to act on, and how coordination is scoped). Accordingly, maintaining policies becomes a form of distributed programming, with implicit assumptions about spatial structure and authority boundaries that may be fragile under change.

There is also much social and spatial ambiguity in real-world settings. Inferring whether users should be linked (implying interaction) or simply co-located (with no interaction) can be uncertain. Thus, any real-world representation must cope with noisy and uncertain data, while supporting policies that reflect implicit social intent.

6 Conclusions

We propose a spatial representation for networked environments with bigraphs, enabling structured reasoning over space, connectivity, and context. By explicitly modeling spatial hierarchy and logical associations, bigraphs offer a foundation for agents that act locally, respect privacy boundaries, and scale across environments. While realizing our framework involves addressing issues around capturing social and network relationships and supporting robust distributed coordination, it lays a strong foundation for responsive, interpretable, and spatially-grounded agent infrastructures.

References

[1] Ebtihal Althubiti, Michele Sevegnani, and Archibald Blair. 2025. Formalising Privacy Regulations with Bigraphs. v1 (2025). https://eprints.gla.ac.uk/353722/

[2] Anthropic. 2025. https://modelcontextprotocol.io/introduction Accessed: 2025-06-29.

[3] Apple Inc. 2022. RoomPlan API. https://developer.apple.com/roomplan/. Accessed: 2025-06-17.

[4] Junjie Chen, Haitao Li, Jingli Yang, Yiqun Liu, and Qingyao Ai. 2025. Enhancing LLM-Based Agents via Global Planning and Hierarchical Execution. *arXiv preprint arXiv:2504.16563* (2025).

[5] Alibaba Cloud. 2024. Qwen3: A family of open-source language models. https://github.com/QwenLM/Qwen. Accessed: 2025-06-17.

[6] Ryan Gibb, Anil Madhavapeddy, and Jon Crowcroft. 2023. Where on Earth is the Spatial Name System?. In *Proceedings of the 22nd ACM Workshop on Hot Topics in Networks* (Cambridge, MA, USA) *(HotNets '23)*. Association for Computing Machinery, New York, NY, USA, 79–86. https://doi.org/10.1145/3626111.3628210

[7] Google. 2016. Google Home. https://store.google.com/us/product/google_home. Accessed: 2025-06-12.

[8] Song Han, Huizi Mao, and William J. Dally. 2016. Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding. arXiv:1510.00149 [cs.CV]

[9] Home Assistant Community. 2025. Home Assistant: Open source home automation that puts local control and privacy first. https://www.home-assistant.io. Accessed: 2025-06-17.

[10] Yiping Kang, Johann Hauswald, Cao Gao, Austin Rovinski, Trevor Mudge, Jason Mars, and Lingjia Tang. 2017. Neurosurgeon: Collaborative Intelligence Between the Cloud and Mobile Edge. *SIGPLAN Not.* 52, 4 (apr 2017), 615–629. https://doi.org/10.1145/3093336.3037698

[11] Evan King, Haoxiang Yu, Sangsu Lee, and Christine Julien. 2024. Sasha: Creative Goal-Oriented Reasoning in Smart Homes with Large Language Models. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 8, 1 (March 2024), 1–38. https://doi.org/10.1145/3643505

[12] Raghuraman Krishnamoorthi. 2018. Quantizing deep convolutional networks for efficient inference: A whitepaper. arXiv:1806.08342 [cs.LG]

[13] Qinbin Li, Junyuan Hong, Chulin Xie, Jeffrey Tan, Rachel Xin, Junyi Hou, Xavier Yin, Zhun Wang, Dan Hendrycks, Zhangyang Wang, Bo Li, Bingsheng He, and Dawn Song. 2024. LLM-PBE: Assessing Data Privacy in Large Language Models. arXiv:2408.12787 [cs.CR] https://arxiv.org/abs/2408.12787

[14] Ji Lin, Wei-Ming Chen, Yujun Lin, John Cohn, Chuang Gan, and Song Han. 2020. MCUNet: Tiny Deep Learning on IoT Devices. *CoRR* abs/2007.10319 (2020). arXiv:2007.10319 https://arxiv.org/abs/2007.10319

[15] Yaxi Lu, Haolun Li, Xin Cong, Zhong Zhang, Yesai Wu, Yankai Lin, Zhiyuan Liu, Fangming Liu, and Maosong Sun. 2025. Learning to Generate Structured Output with Schema Reinforcement Learning. arXiv:2502.18878 [cs.CL] https://arxiv.org/abs/2502.18878

[16] Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. 2018. ShuffleNet V2: Practical Guidelines for Efficient CNN Architecture Design. arXiv:1807.11164 [cs.CV]

[17] Anil Madhavapeddy, K C Sivaramakrishnan, Gemma Gordon, and Thomas Gazagnaire. 2018. An architecture for interspatial communication. In *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. 716–723. https://doi.org/10.1109/INFCOMW.2018.8406931

[18] Alessio Mansutti, Marino Miculan, and Marco Peressotti. 2014. Multi-agent Systems Design and Prototyping with Bigraphical Reactive Systems. In *Proceedings of the 14th IFIP WG 6.1 International Conference on Distributed Applications and Interoperable Systems - Volume 8460*. Springer-Verlag, Berlin, Heidelberg, 201–208. https://doi.org/10.1007/978-3-662-43352-2_16

[19] Josh Millar, Yushan Huang, Sarab Sethi, Hamed Haddadi, and Anil Madhavapeddy. 2025. Benchmarking Ultra-Low-Power μNPUs. arXiv:2503.22567 [cs.LG] https://arxiv.org/abs/2503.22567

[20] Robin Milner. 2008. The Space and Motion of Communicating Agents. (December 2008). https://www.cl.cam.ac.uk/archive/rm135/Bigraphs-draft.pdf Unpublished manuscript, University of Cambridge.

[21] openHAB Community. 2010. openHAB: Empowering the Smart Home. https://www.openhab.org. Accessed: 2025-06-12.

[22] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. 2016. XNOR-Net: ImageNet Classification Using Binary Convolutional Neural Networks. arXiv:1603.05279 [cs.CV]

[23] Dmitriy Rivkin, Francois Hogan, Amal Feriani, Abhisek Konar, Adam Sigal, Steve Liu, and Greg Dudek. 2024. SAGE: Smart home Agent with Grounded Execution. arXiv:2311.00772 [cs.AI] https://arxiv.org/abs/2311.00772

[24] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. 2019. MobileNetV2: Inverted Residuals and Linear Bottlenecks. arXiv:1801.04381 [cs.CV]

[25] Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language Models Can Teach Themselves to Use Tools. *arXiv preprint arXiv:2302.04761* (2023).

[26] Michele Sevegnani and Muffy Calder. 2016. BigraphER: Rewriting and Analysis Engine for Bigraphs. In *Computer Aided Verification - 28th International Conference, CAV 2016, Toronto, ON, Canada, July 17-23, 2016, Proceedings, Part II*. 494–501. https://doi.org/10.1007/978-3-319-41540-6_27

[27] Leming Shen, Qiang Yang, Xinyu Huang, Zijing Ma, and Yuanqing Zheng. 2025. GPIoT: Tailoring Small Language Models for IoT Program Synthesis and Development. arXiv:2503.00686 [cs.SE] https://arxiv.org/abs/2503.00686

[28] Yihe Tang, Wenlong Huang, Yingke Wang, Chengshu Li, Roy Yuan, Ruohan Zhang, Jiajun Wu, and Li Fei-Fei. 2025. UAD: Unsupervised Affordance Distillation for Generalization in Robotic Manipulation. arXiv:2506.09284 [cs.RO] https://arxiv.org/abs/2506.09284

[29] Guanghan Wu, Sasu Tarkoma, and Roberto Morabito. 2025. Consolidating TinyML Lifecycle with Large Language Models: Reality, Illusion, or Opportunity? arXiv:2501.12420 [cs.SE] https://arxiv.org/abs/2501.12420

[30] Jihan Yang, Shusheng Yang, Anjali W. Gupta, Rilyn Han, Li Fei-Fei, and Saining Xie. 2024. Thinking in Space: How Multimodal Large Language Models See, Remember, and Recall Spaces. arXiv:2412.14171 [cs.CV] https://arxiv.org/abs/2412.14171

[31] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. ReAct: Synergizing Reasoning and Acting in Language Models. *arXiv preprint arXiv:2210.03629* (2023).

[32] Christophe El Zeinaty, Wassim Hamidouche, Glenn Herrou, Daniel Menard, and Merouane Debbah. 2025. Can LLMs Revolutionize the Design of Explainable and Efficient TinyML Models? arXiv:2504.09685 [cs.LG] https://arxiv.org/abs/2504.09685