

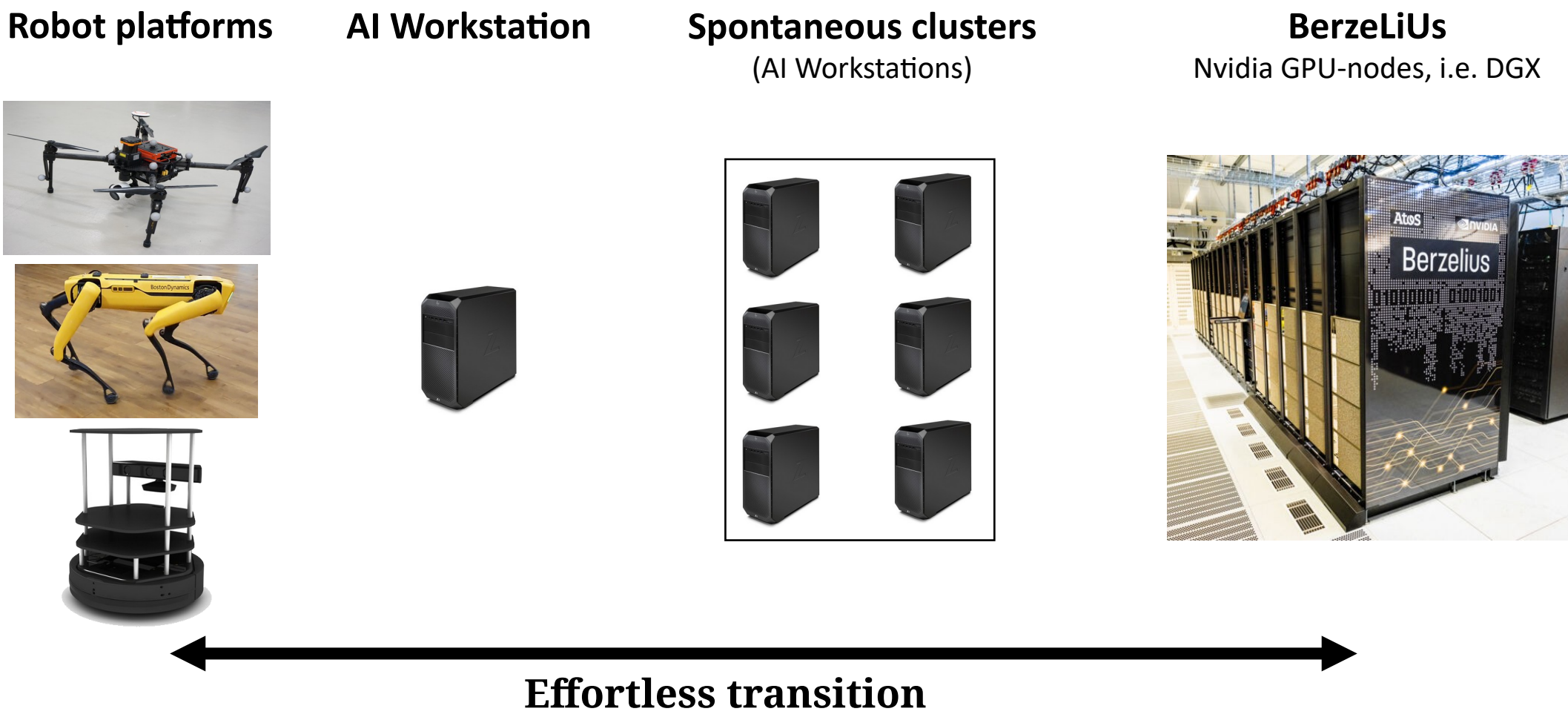
Scalable AIOps: Reproducible AI Research Across Every Scale

Mattias Tiger, Niklas Wretblad, Daniel de Leng, Tohid Ardeshiri, Fredrik Heintz

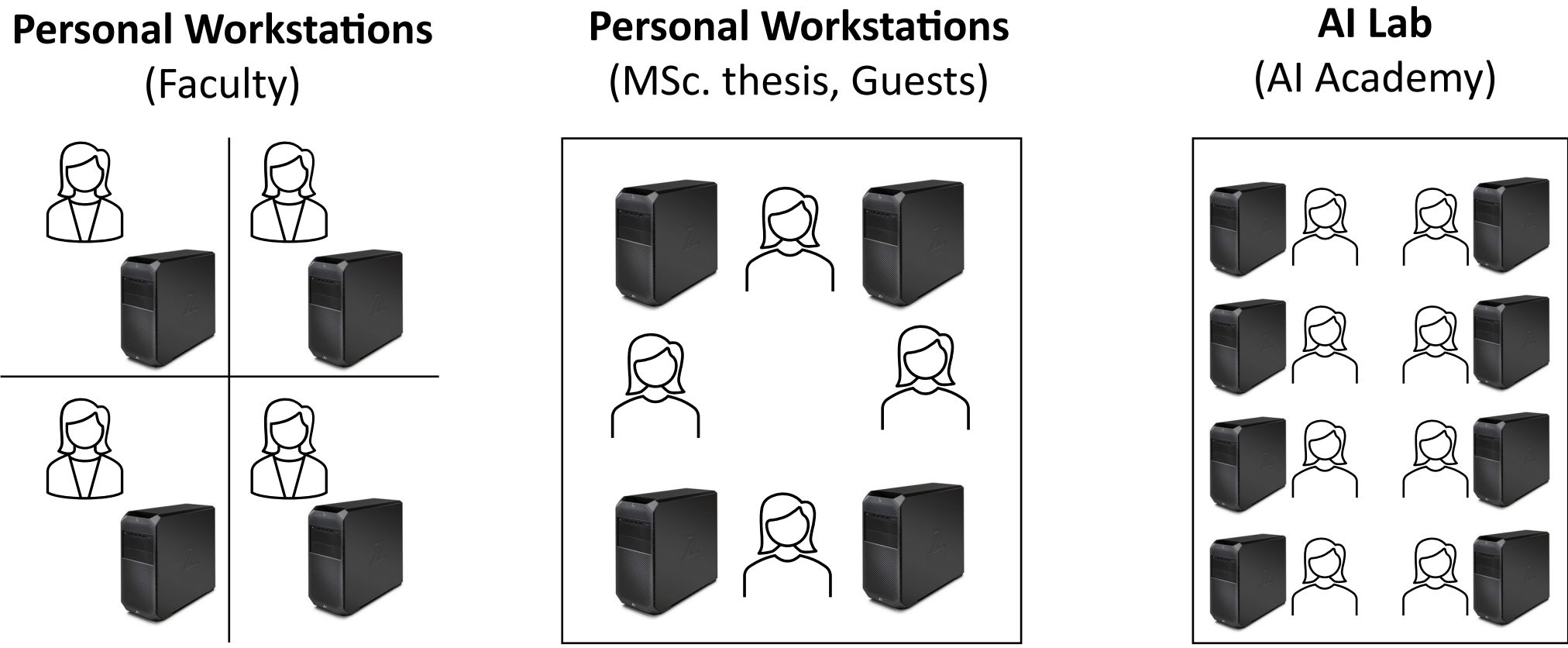
Artificial Intelligence, Machine Learning, and Robotics research are now limited less by ideas than by infrastructure. Experiments often outgrow personal GPU workstations and must scale to national compute resources. Scalable AIOps addresses this gap by enabling reproducible, collaborative, and efficient AI research across heterogeneous hardware—from single workstations to large-scale clusters.

From Local Experiments to Large-Scale AI Research

AI and ML research are advancing faster than our ability to manage experiments. Projects that start on a single GPU workstation must often scale to national compute facilities, yet reproducibility and collaboration remain difficult.



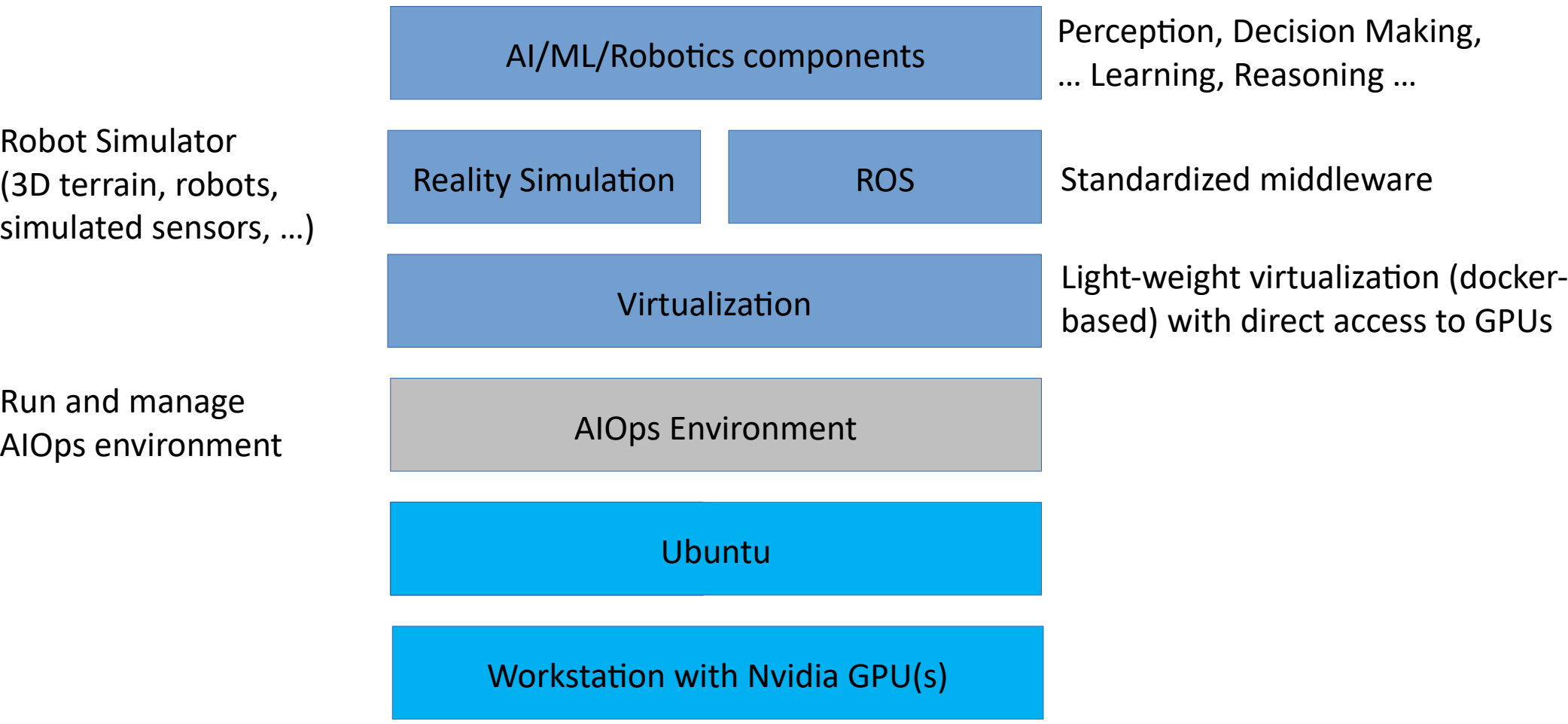
Scalable AIOps develops a *scientific DevOps environment* for ELLIIT researchers—combining containerization, detailed version control, and lightweight virtualization. It bridges **personal workstations**, **shared lab clusters** such as AI Academy’s **Stellar**, and national supercomputing resources like **BerzeLiUs**, creating a continuous and reproducible research pipeline. *The same experiment, rebuilt identically anywhere—from laptop to data-center.*



AIOps DevOps Stack – From Code to Runtime

- To manage complex AI/ML/Robotics systems—from perception and reasoning to control—AIOps uses a layered DevOps workflow where every component is version-tracked and portable:
- **Everything under version control:** code, data, models, environments.
 - **Reproducible execution:** Docker & Docker Compose encapsulate build and runtime.
 - **Integrated simulation:** ROS and virtual sensors run inside containers.
 - **GPU efficiency:** lightweight virtualization with direct GPU access.
 - **Seamless transition:** identical stacks across local labs and national clusters.

Reproducibility increases as every layer—from source code to runtime OS—is tracked and shareable



Version Control is Key

Reproducibility increases ↓	• Source code	(example) (main.py)	How? GIT
	• Data		GIT / GIT LFS
	• Data sets / Models		GIT LFS
	• Libraries and packages	(numpy)	Requirements.txt
	• Runtime libraries	(CUDA)	Container / Conda
	• Build environment	(gcc, ament)	Container
	• Runtime environment	(Ubuntu)	Container
	• Development environment	(VSCode, Cursor, .bashrc)	Container

Stellar Environment – A Living Implementation

At Linköping University’s AI Academy, the Stellar system embodies AIOps in practice. Dozens of AI workstations are connected through a secure VPN mesh, forming a shared, reproducible infrastructure for AI research and education. Stellar supports master’s theses, advanced project courses, and summer schools such as WARA-PS on autonomous systems for public safety.



Tools and Environments for

- Rapid prototyping
- Reproducible execution
- Traceable changes
- Flexible operations
- Limited and localized destruction
- Automated full system tests (scenarios)

Each workstation runs **rootless Docker**, allowing students and staff to install and experiment freely while remaining compliant with university IT policy. Multiple users share machines safely, with all environments version-controlled. Stellar unites separate AI labs into a policy-compliant, flexible research and education platform.

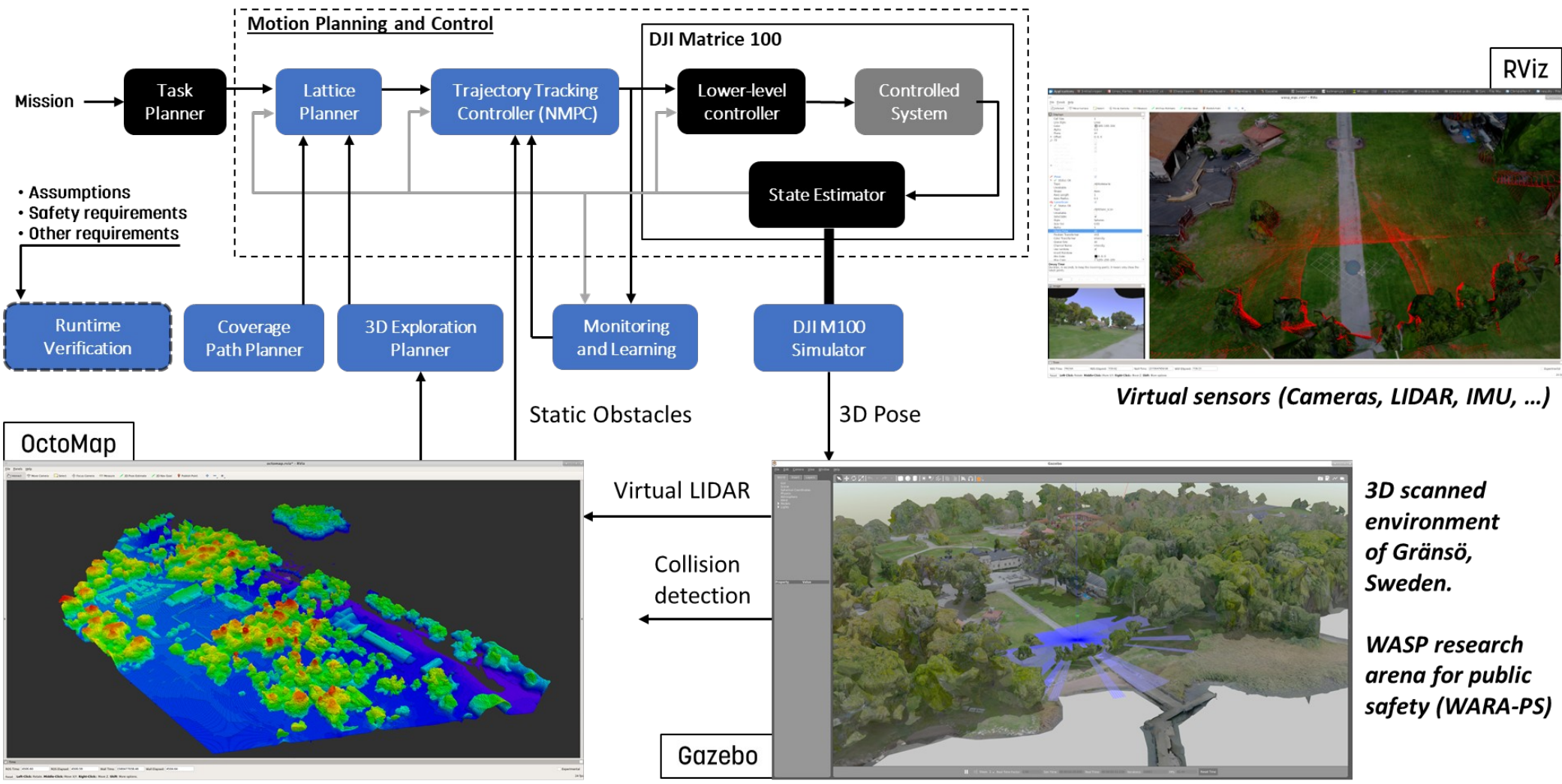
Spontaneous Cluster – Dynamic Scaling at the Edge

Within the Stellar environment, AIOps adds a dynamic layer: the spontaneous cluster. Each workstation monitors user activity and availability. When idle, it joins a distributed computation pool; when a user logs in, it halts active jobs to free resources.

Lightweight tools such as Psionic Nonsense coordinate these transitions over the Stellar VPN mesh. Dockerized jobs and shared volumes keep all nodes synchronized and reproducible, bridging the gap between individual workstations and large-scale clusters like BerzeLiUs.

Example AI-Robotics Stack and Simulation Environment

A fully version-controlled AI-robotics stack—including ROS, Gazebo, RViz, OctoMap, and all AI components—runs entirely in Docker. Every layer, from build scripts to simulation worlds, is portable and reproducible across systems.



Stacks developed over five years ago still run unchanged today, demonstrating AIOps-driven longevity and scientific reuse. *Complex robotic systems remain runnable and reproducible — years after development.*