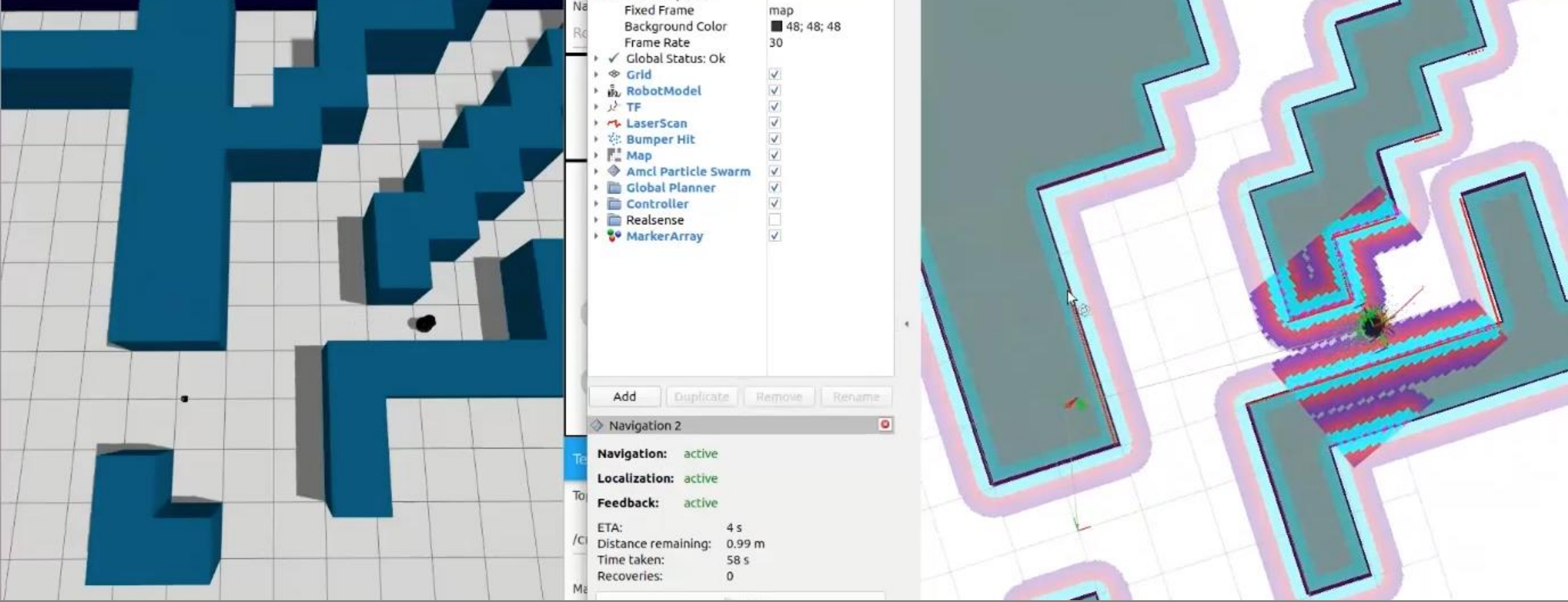

Robotics & XR

5 ECTS
Ilkka Jormanainen and Samuel Yigzaw
{firstname.lastname}@uef.fi

AI in Robotics
November 25, 2024

Theme 5: Robotics and AI

- *Week 48 (November 25 – December 1)*
- AI methods in robotics
- Machine vision
- Neural networks
- Speech recognition
- Reinforcement and model learning

AI methods and applications in robotics

- Robotics and AI are often considered as synonyms
- Almost all current robotics solutions have AI driven features
  - Machine learning for navigation tasks
  - Machine vision for object recognition and manipulation
  - NLP and LLMs for human-robot interaction
- New and emerging technologies like LLMs and edge computing will change how robots are developed and operated

Challenges with AI applications in robotics

- Data challenges
  - Quality and quantity of training data
  - Sim-to-real gap
  - Real-time data processing
- Computational constrains
  - Limited resources
  - Latency and bandwidth
- Integration
  - Learning vs. controlling
  - Sensor fusion and synchronizing multiple data sources
  - Generalization of AI models

Challenges with AI applications in robotics

- Safety and reliability
  - Uncertainty in predictions – for instance road signs for self-driving cars
- Failure recovery
  - Robots must recover automatically, but many AI models inherently lack robustness against unexcepted scenarios
- Real-world risks
  - Physical robots may cause lots of harm if AI goes wrong (compared to software-AI)

Challenges with AI applications in robotics

- Ethical and social challenges
  - Transparency and explainability – black boxes
  - Biased training data – Unwanted or unsafe behavior of the robot
  - Job displacement and societal impacts
- Deployment and maintenance
  - Adaptation to new or dynamic environments
  - Software-hardware mismatch
  - Scalability

How to solve challenges?

- Model optimization – pruning, quantization
- Domain adaptation – fine-tuning or transfer learning
- Safety frameworks – rule-based or human-driven safety measures
- Hybrid architecture – edge computing to balance between on-board and cloud
- Explainable AI – making users to better understand the models and how to interpret them

Early Days: Rule-Based Systems and Logical AI (1950s–1970s)

- **AI Foundations**:
  - Alan Turing's *"Computing Machinery and Intelligence"* (1950) and the proposal of the Turing Test inspired early thoughts on AI's potential for robotics.
  - Early robotics relied on rule-based systems and explicit programming to perform tasks in controlled environments.
- **Example**: Shakey the Robot (1966–1972), developed by SRI International, was the first general-purpose robot to combine problem-solving AI with perception and navigation. It relied on logical reasoning to plan actions.

Emergence of Machine Learning (1980s–1990s)

- **From Rules to Learning**:
  - AI shifted from rule-based systems to machine learning (ML), where robots could learn patterns from data rather than relying solely on pre-defined instructions.
  - Early neural networks, though limited by computational power, laid the groundwork for modern AI.
- **Example**: Autonomous land vehicle projects like CMU's ALV (1980s) showcased how early computer vision techniques and ML could help robots navigate in outdoor environments.

Evolution of Probabilistic Robotics (1990s–2000s)

- **Bayesian Inference and SLAM**:
  - Probabilistic approaches (e.g., Markov models, Kalman filters) enabled robots to deal with uncertainty in real-world environments.
  - Simultaneous Localization and Mapping (SLAM) algorithms emerged, allowing robots to map unknown environments while tracking their position.
- **Example**: The Mars rovers (*Spirit* and *Opportunity*, 2004) utilized AI for navigation, decision-making, and self-diagnosis based on probabilistic reasoning.

Deep Learning Revolution (2010s)

- **Advances in Neural Networks**:
  - The resurgence of neural networks, powered by deep learning, brought transformative changes to robotics.
  - Robots could now process vast amounts of sensory data (e.g., images, speech) with unprecedented accuracy.
- **Example**:
  - Boston Dynamics' robots began leveraging AI for dynamic motion control.
  - NVIDIA's use of deep reinforcement learning enabled autonomous drones and cars to navigate complex environments.

Integration of AI Across Disciplines (2020s and Beyond)

- **Collaborative AI and Multi-Agent Systems**:
  - AI now enables robots to collaborate with humans and other robots in shared environments.
- **Edge Computing and Federated Learning**:
  - Robots can process AI algorithms locally, enabling faster and more autonomous decision-making.
- **Example**:
  - Tesla’s AI-driven autonomous driving systems demonstrate integration of computer vision, deep learning, and real-time decision-making.
  - Robotic assistants like Pepper or Sophia showcase the combination of NLP, computer vision, and advanced learning algorithms.

Case 1: Boston Dynamics – AI-Driven Agility

- Boston Dynamics is renowned for robots like *Spot* (quadruped) and *Atlas* (bipedal), showcasing human-like agility and complex movement capabilities.
- Handling high degrees of freedom in robot kinematics through hierarchical controllers.
- Seamlessly combining learned behaviors (RL) with classical control strategies for optimal performance.

Image source: Helsingin Sanomat / Joe Raedle / AFP

Case 1: Boston Dynamics’ robots

- **Dynamic Motion Planning**
  - Use of Model Predictive Control (MPC) to optimize real-time motion trajectories based on robot dynamics and constraints.
  - Combination of trajectory optimization with contact dynamics (foot placement and force distribution).
- **Reinforcement Learning (RL)**
  - RL is employed to train robots in simulated environments, significantly reducing the risk of hardware damage.
  - Deep RL frameworks (e.g., Proximal Policy Optimization, PPO) are used to enable balance and adaptive recovery from disturbances.
- **Computer Vision**
  - Visual SLAM and depth perception systems enable obstacle avoidance and mapping.
  - Integration of LiDAR and stereo cameras for robust environmental understanding.

Case Study 2: Agricultural Robotics – AI for Fruit Picking

- Robotic arms equipped with vision systems are designed to autonomously pick fruits like apples and strawberries.
- Environmental variability (lighting, occlusion) addressed with data augmentation and domain adaptation techniques.
- Real-time decision-making achieved through edge computing platforms like NVIDIA Jetson.

Image source: Monash University

Case 2: Fruit picking robot

- **Computer Vision**
  - Use of Convolutional Neural Networks (CNNs) for fruit detection and classification.
  - Segmentation models (e.g., U-Net or Mask R-CNN) for precise identification of harvestable fruits.
- **Grasp Planning**
  - Algorithms based on deep learning (e.g., GraspNet) to compute optimal grasp points based on fruit geometry and pose.
  - Integration with motion planning algorithms like Rapidly-Exploring Random Trees (RRT) or Probabilistic Roadmaps (PRM) for collision-free arm trajectories.
- **Reinforcement Learning for Gripper Control**
  - Adaptive control strategies to fine-tune the force applied during fruit picking to prevent damage.

Case 3: Warehouse Robotics – Amazon's Kiva Systems

- Autonomous mobile robots (AMRs) streamline warehouse operations by moving inventory shelves to human workers.
- Scalability: Distributed computing ensures that the system can manage thousands of robots.
- Latency: Low-latency wireless communication protocols (e.g., 5G, Zigbee) reduce delays in command transmission.

Image source: Amazon

Case 2: Amazon warehouse robots

- **Multi-Agent Path Planning (MAPF)**
  - Algorithms like Cooperative A* and prioritized planning ensure efficient pathfinding and collision avoidance in multi-robot systems.
  - Use of real-time updates for dynamic re-routing based on warehouse activity.
- **Localization and Mapping**
  - Kiva robots employ fiducial markers (e.g., QR codes) for precise localization within a grid-based environment.
  - Algorithms like Extended Kalman Filter (EKF) refine positional estimates using odometry and sensor data.
- **Task Allocation**
  - Centralized systems use task scheduling algorithms (e.g., Hungarian algorithm) to allocate robots efficiently.
  - Optimization techniques (e.g., Integer Linear Programming) balance workload and minimize idle time.

LLMs and robotics

- Human-robot interaction
  - Natural language commands and interaction
  - Service and social robots, healthcare, robot companions
- Instruction parsing and task execution
  - LLMs to interpret vague commands and translate them into actionable tasks
  - RPA applications – tasks that require decision making and context understanding
- Multi-modal learning and reasoning
  - Multimodal LLMs may help robotics applications to combine language and vision into action
  - Interpretation of visual data in the surrounding

LLMs and robotics

- Knowledge retrieval for decision making
    - Robotics applications may access vast amount of knowledge while executing tasks -> decision making based on newly adopted information
- LLMs may be combined with knowledge graphs or retrieval-augmented generation (RAG) systems enable robots to provide accurate, context-aware responses, even in niche technical domains.

LLMs and robotics

- LLMs assist in programming robots by generating control code based on natural language descriptions or demonstrations.
- Applications: Rapid prototyping of robotic behaviors or teaching robots to replicate complex actions.
- For example: Codex (OpenAI’s model) can write Python code for controlling robotic arms or designing algorithms for specific tasks based on user-provided prompts.

Challenges with LLMs

- Real-Time Processing
  - LLMs often require significant computational resources, which may conflict with the real-time processing needs of robotics applications. Edge AI or optimization methods are critical to overcoming this.
- Grounding and Context Awareness
  - LLMs lack physical grounding; hence, errors can occur when translating abstract instructions into physical actions. Combining LLMs with sensory data and reinforcement learning helps address this.
- Ethical Concerns
  - Risks associated with misunderstanding or over-reliance on language-based decisions in safety-critical applications.

Edge computing in robotics

- Data is processed locally on the robot or nearby devices (e.g., on-premises servers or edge gateways) instead of relying entirely on cloud-based resources
- Crucial for robotics due to the real-time and often safety-critical nature of robotic tasks
- Low Latency for Real-Time Decision-Making
- Bandwidth Optimization
- Enhanced Reliability
- Data Privacy and Security

Edge computing for robotics

- Perception Systems
  - Real-time object detection, tracking, and recognition using edge AI models (e.g., YOLO or MobileNet) enable robots to navigate complex environments.
  - Example: Autonomous delivery robots use edge vision systems to identify obstacles and pedestrians in real-time.
- Autonomous Navigation
  - Algorithms like Visual SLAM (Simultaneous Localization and Mapping) and path planning often rely on edge computing to map environments and localize the robot without delays.
- Natural Language Processing (NLP)
  - Lightweight LLMs or speech recognition models on edge devices allow robots to interact conversationally without relying on cloud APIs, reducing latency.

Edge computing for robotics

- Collaborative Robotics (Cobots)
  - Edge computing to support multi-agent coordination in warehouses or assembly lines, enabling robots to communicate and plan actions collectively without requiring centralized cloud systems.
- Predictive Maintenance
  - AI models running at the edge analyze sensor data from robotic components (e.g., motors, joints) to predict and prevent failures before they occur.

Challenges with edge AI in robotics

- **Hardware Constraints**
  - Edge devices often have limited compute power and storage compared to cloud systems, requiring highly optimized AI models (e.g., pruning, quantization).
- **Energy Efficiency**
  - Power consumption is critical, especially for mobile robots like drones or autonomous vehicles, where battery life is limited.
- **Integration Complexity**
  - Balancing computational loads between edge devices and cloud systems can complicate system design.

Some edge AI platforms for robotics

- **NVIDIA Jetson Series**
  - Compact GPU-based platforms designed for AI tasks on the edge. Widely used in drones, autonomous vehicles, and industrial robots.
- **Google Coral**
  - Specialized for low-power AI applications with Tensor Processing Units (TPUs). Used in robots for image classification and edge inferencing.

Some edge AI platforms for robotics

- **Intel Movidius**
  - Offers hardware acceleration for computer vision tasks, allowing robots to process complex imagery locally.
- **TinyML**
  - Brings AI capabilities to edge devices, such as low-cost and low-power microcontrollers (for instance, Arduino TinyML Kit)
  - ML frameworks optimized for microcontrollers, for instance TensorFlow Lite and Edge Impulse
  - Models are trained in advance – real-time adaptation is challenging