Practical AI for Autonomous Robots

Day 1: Introduction to Robotic Software and Architectures

Dr. Timothy Wiley

School of Computing Technologies

RMIT University

ESSAI July 2025

RMIT UNIVERSITY

Acknowledgement of Country

RMIT University acknowledges the people of the Woi wurrung and Boon wurrung language groups of the eastern Kulin Nation on whose unceded lands we conduct the business of the University.

RMIT University respectfully acknowledges their Ancestors and Elders, past and present.

RMIT also acknowledges the Traditional Custodians and their Ancestors of the lands and waters across Australia where we conduct our business.

Artwork ‘Luwaytini’ by Mark Cleaver, Palawa

About me: Dr. Timothy Wiley

PhD, Online Learning of Robotic Behaviours,
UNSW Australia

Lecturer, Artificial Intelligence,
School of Computing Technologies
STEM College, RMIT University

Manager,
Artificial Intelligence Innovation Lab

Research Interests:

- Autonomous Robotics
- Reinforcement Learning
- Qualitative Reasoning
- Uncrewed Ariel Systems
- Sim2Real

Robot Soccer

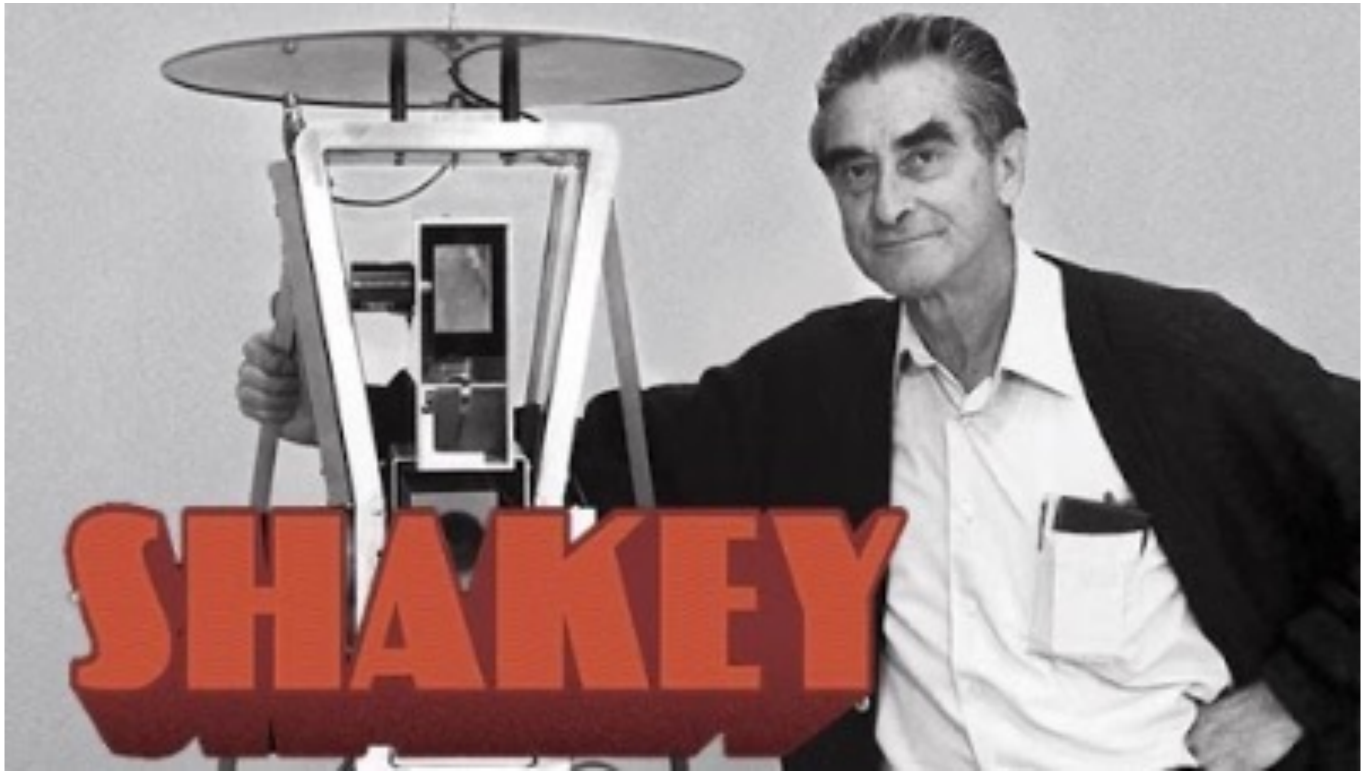
What is Autonomous Robotics?

ESSAI July 2025

What defines an Autonomous Robot?

What defines an autonomous robot to you?

Shakey the Robot



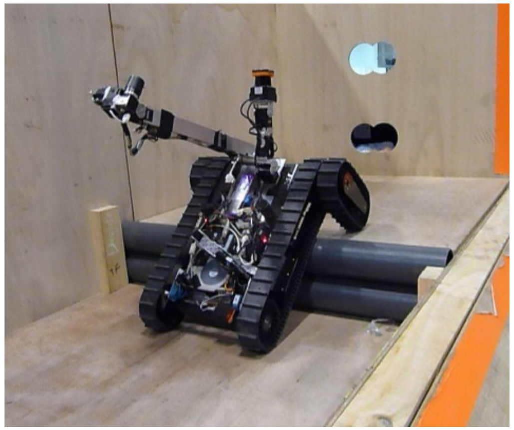
Nilsson, N. J. (1982). Principles of Artificial Intelligence. Berlin Heidelberg: Springer-Verlag

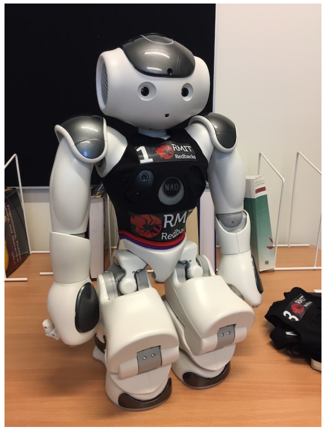
What defines an Autonomous Robot?

Aside from the obvious “act autonomously” that is, “think-and-act for themselves”:

- Perceive the environment
- Interact with the environment
- Internal reasoning

What separates robotics from other fields of AI is the physical input/output





RoboCup Soccer – Aibo

RoboCup Soccer – Aibo

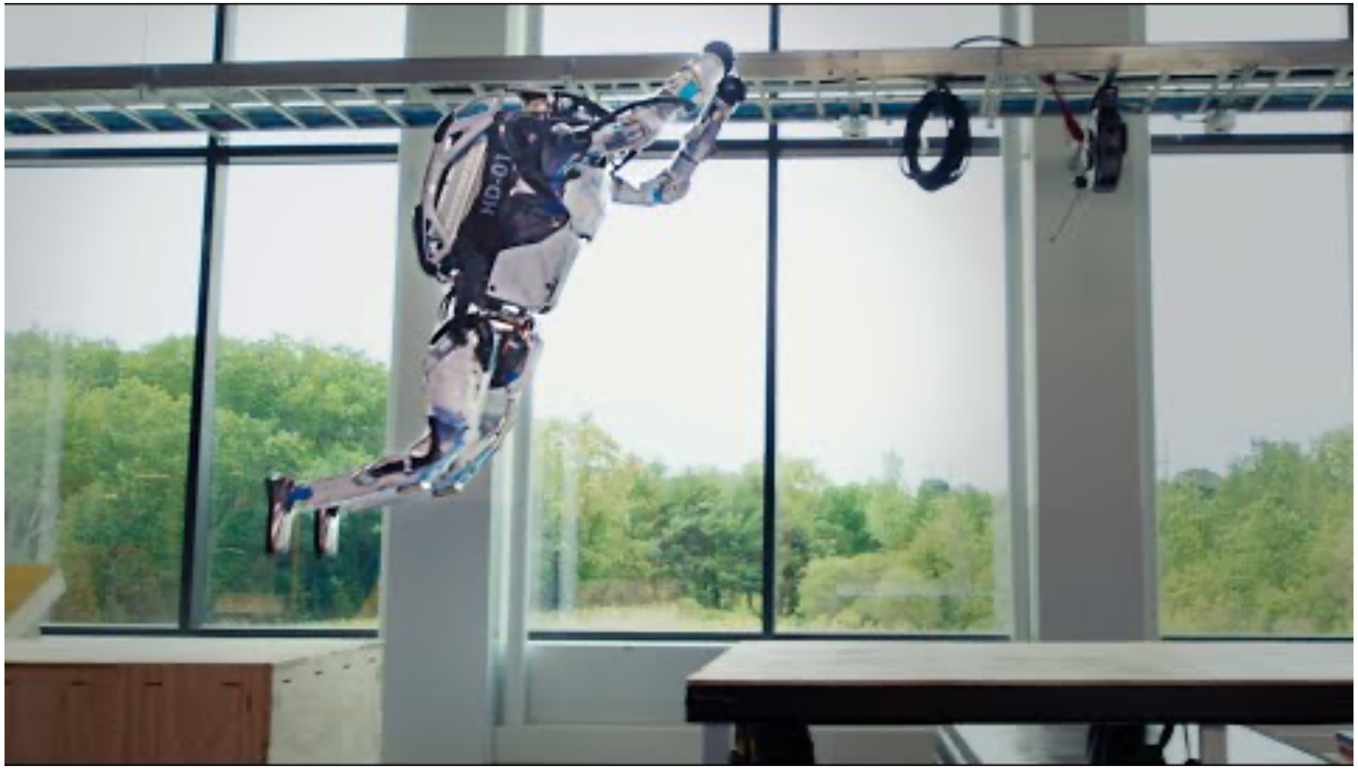
RoboCup Soccer – Nao

RoboCup Soccer – Nao

DARPA Rescue Challenge

DeepMind Learning Soccer

Boston Dynamics ATLAS motion



About the Course

ESSAI July 2025

This Course is about…

- Overview of Autonomous Robotics
- Give a breadth of knowledge rather than a depth
- Provide operational knowledge on autonomous robotic system for later in-depth study
- Applied AI/ML
- Hopefully convince you to work in robotics 😛

This course presumes…

- Some familiarity with common AI techniques such as:
  - Graph algorithms
  - Heuristic Search (A*)
  - Random Sampling
  - Reinforcement Learning fundamentals
  - Machine Learning fundamentals
  - Neural Network structures

Topics

Day 1: Introduction to Robot System + Learning Robot Control

Day 2: Motion Planning and Navigation

Day 3: Localisation and Mapping

Day 4: Robotic Vision

Day 5: Symbolic Task Planning and Planning with LLMs

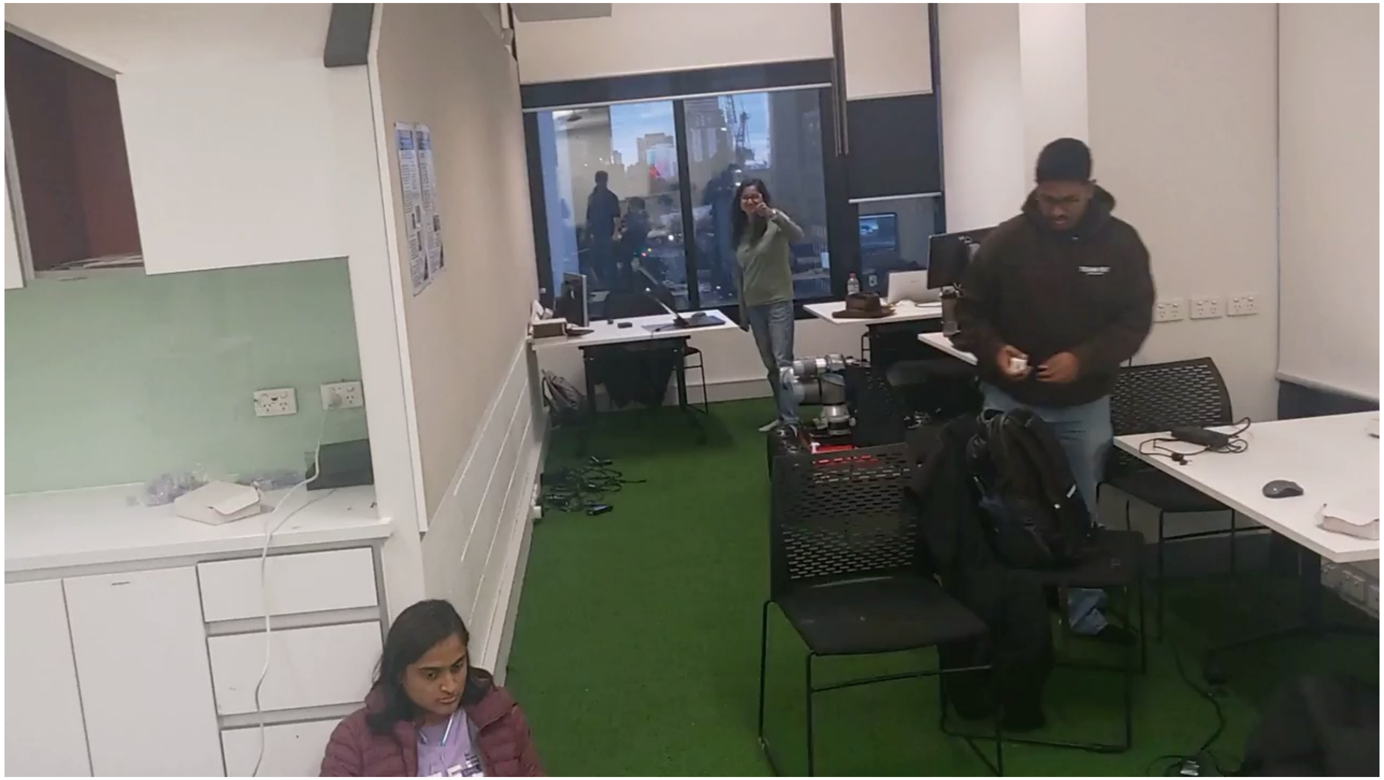
Motivating Example

ESSAI July 2025

Problem: Maze

Problem: Maze

Solution: With AR Visualisation & Control



Solution: To touch each marker

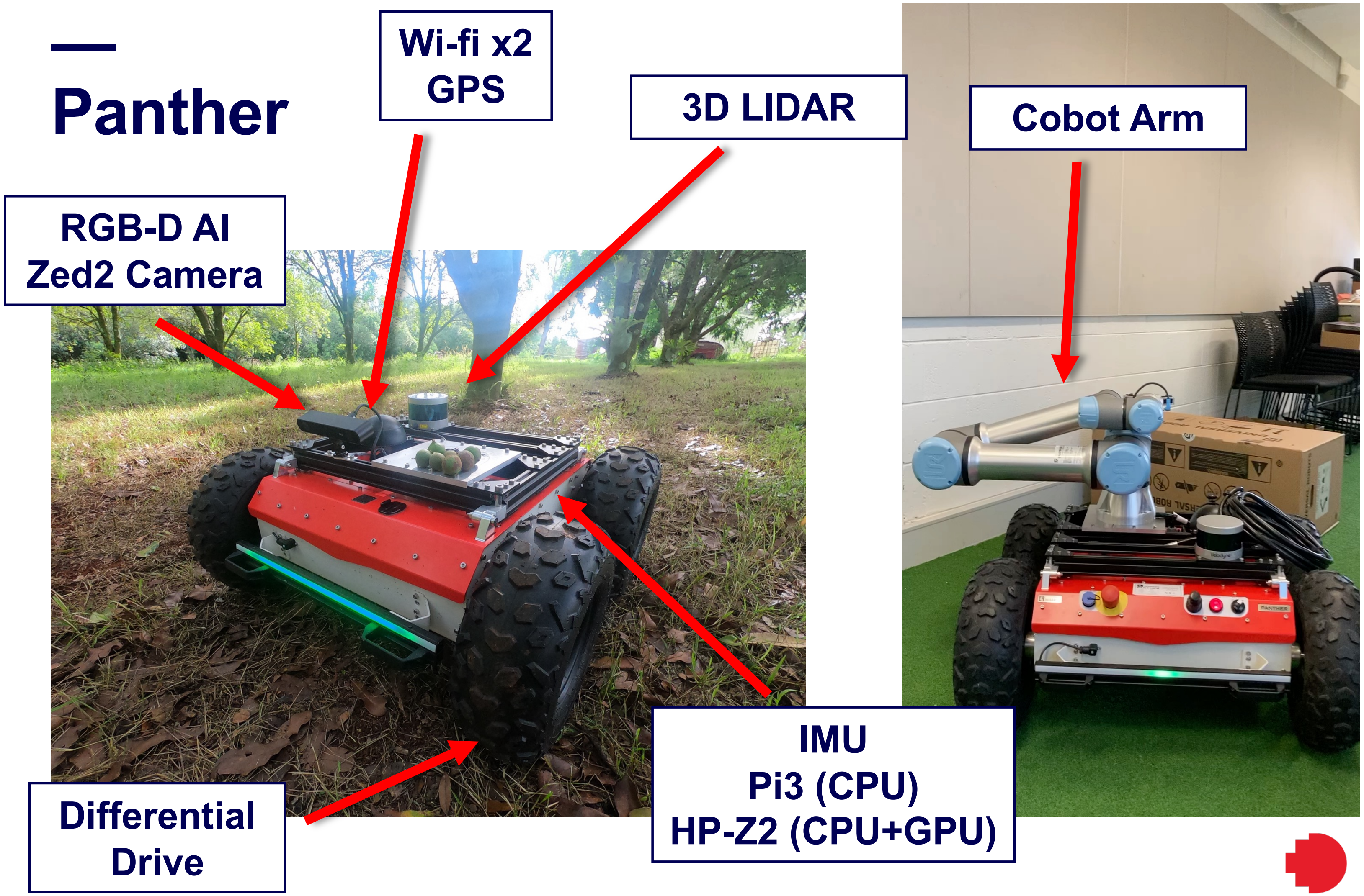
Elements of Robots

ESSAI July 2025

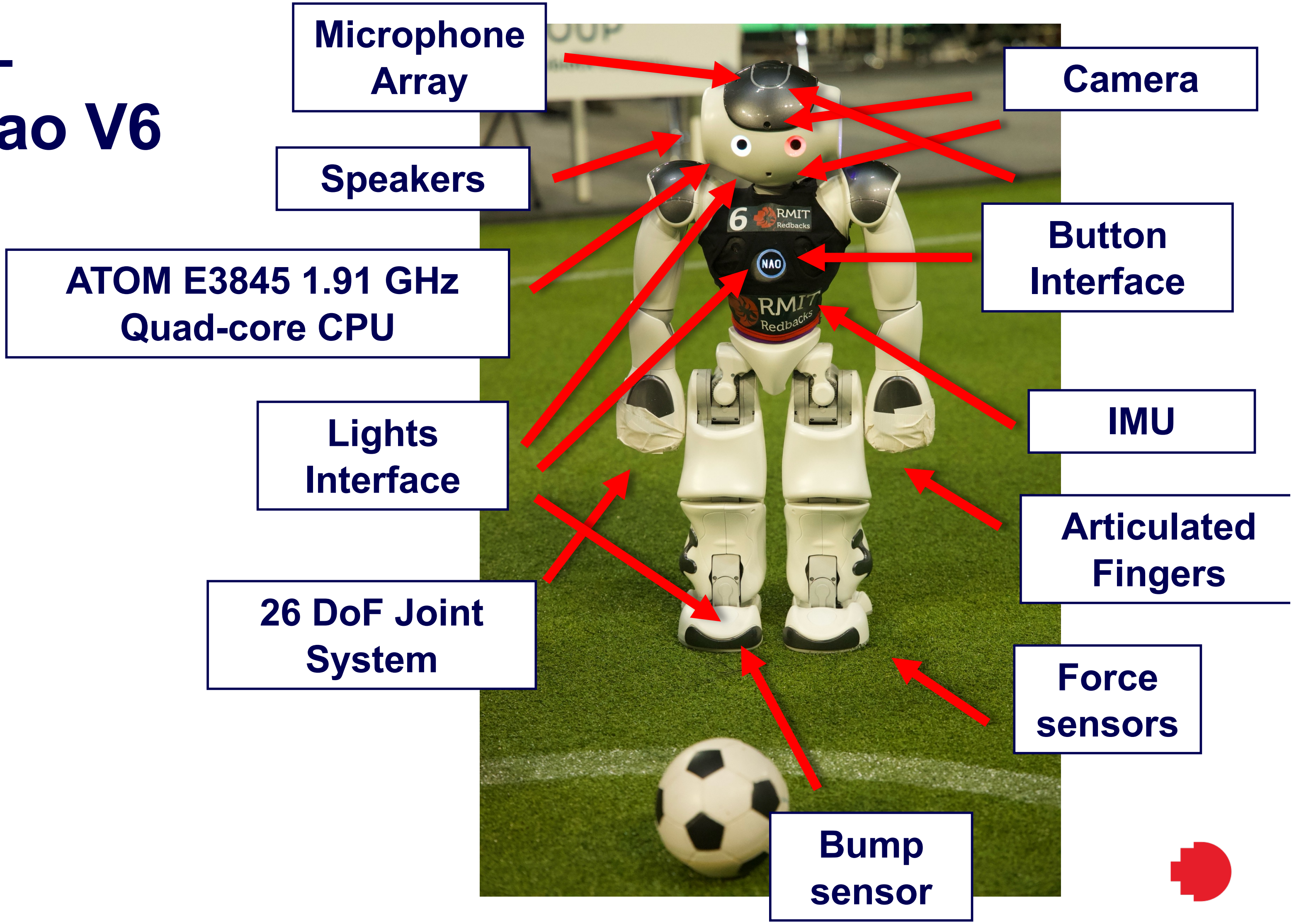
ROSbot


2D LIDAR
RGB-D Camera
Wi-fi
1D Ultrasound
IMU
CPU
Differential or Omnidirectional Drive

Panther


Wi-fi x2
GPS
3D LIDAR
Cobot Arm
RGB-D AI
Zed2 Camera
IMU
Pi3 (CPU)
HP-Z2 (CPU+GPU)
Differential
Drive

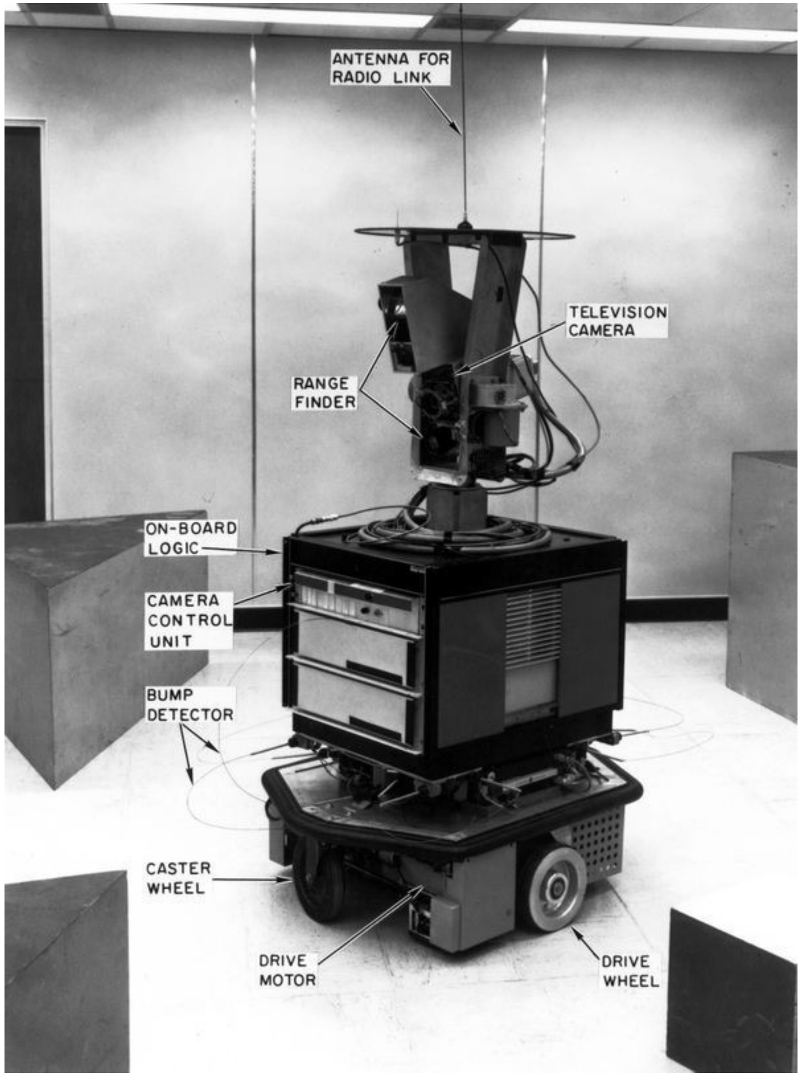
Nao V6



Microphone Array
Camera
Speakers
Button Interface
ATOM E3845 1.91 GHz Quad-core CPU
IMU
Lights Interface
Articulated Fingers
26 DoF Joint System
Force sensors
Bump sensor

Shakey the Robot

Shakey the Robot: The First Robot to Embody Artificial Intelligence



Autonomous Agent Model

ESSAI July 2025

RMIT UNIVERSITY

AI Agent Models

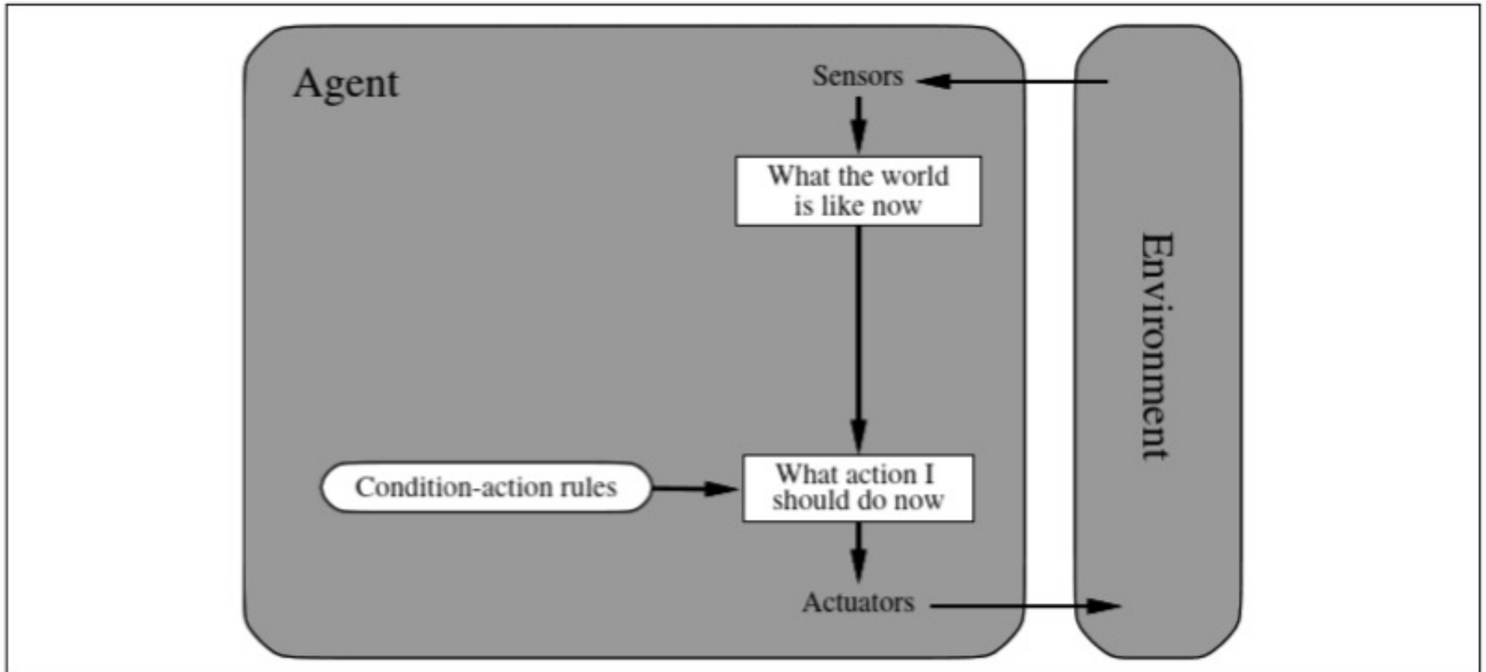


Figure 2.9 Schematic diagram of a simple reflex agent.

Russel & Norvik. Artificial Intelligence: A Modern Approach (2017),

AI Agent Models

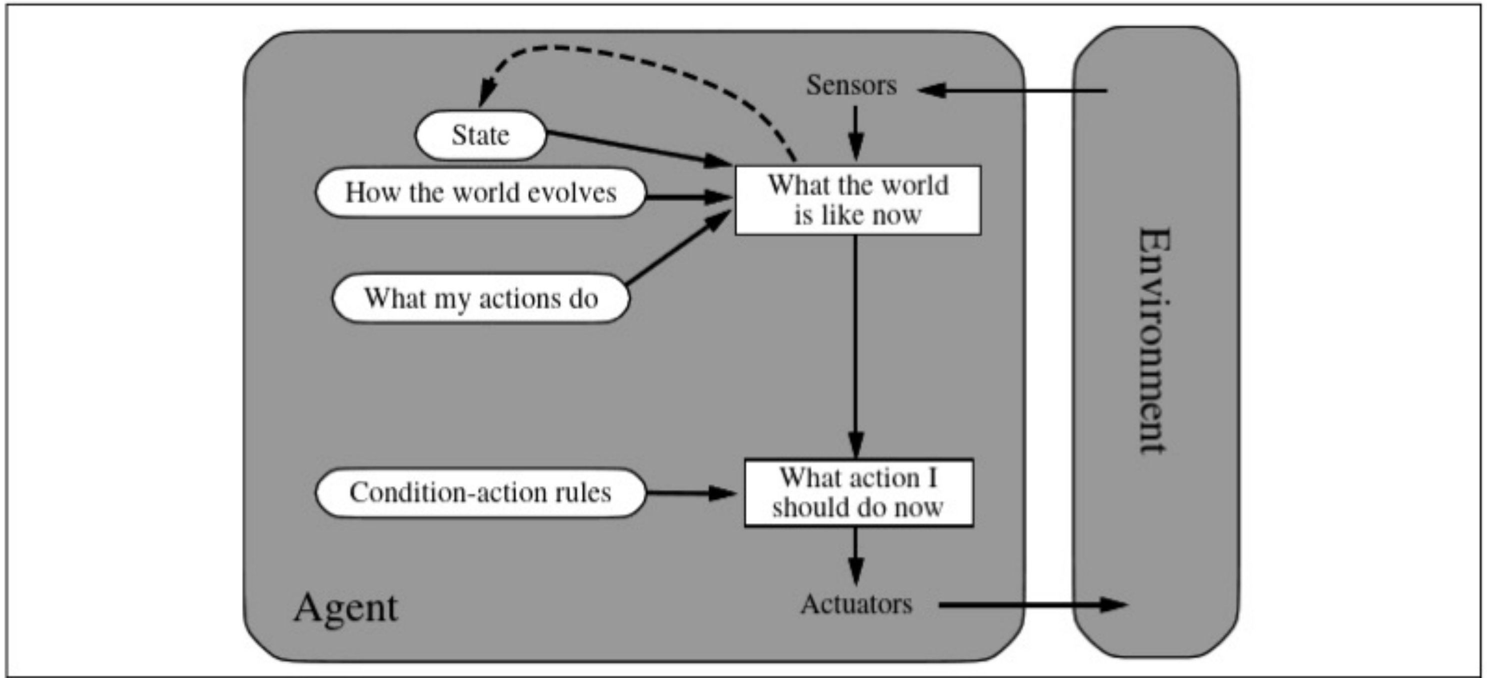


Figure 2.11 A model-based reflex agent.

Russel & Norvik. Artificial Intelligence: A Modern Approach (2017),

AI Agent Models

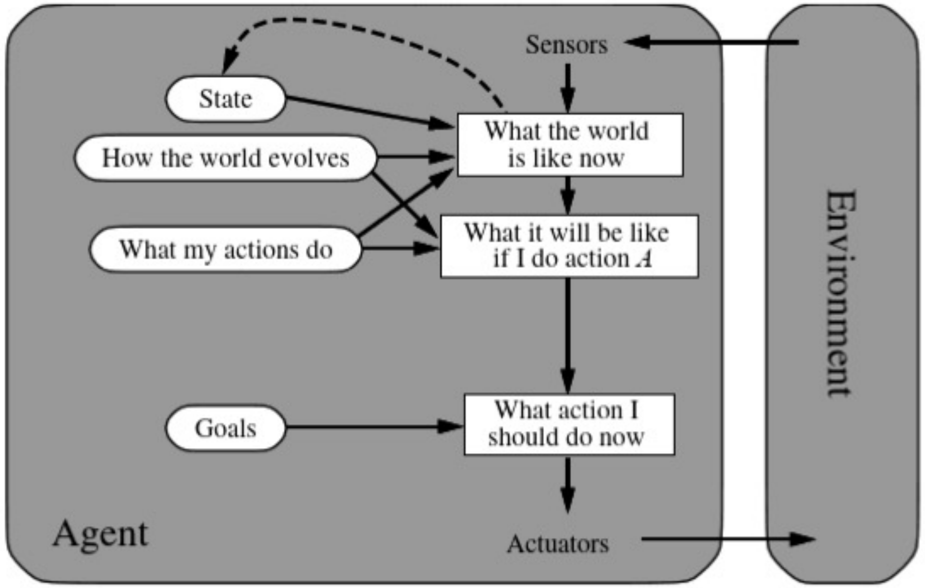


Figure 2.13 A model-based, goal-based agent. It keeps track of the world state as well as a set of goals it is trying to achieve, and chooses an action that will (eventually) lead to the achievement of its goals.

Russel & Norvik. Artificial Intelligence: A Modern Approach (2017),

AI Agent Models

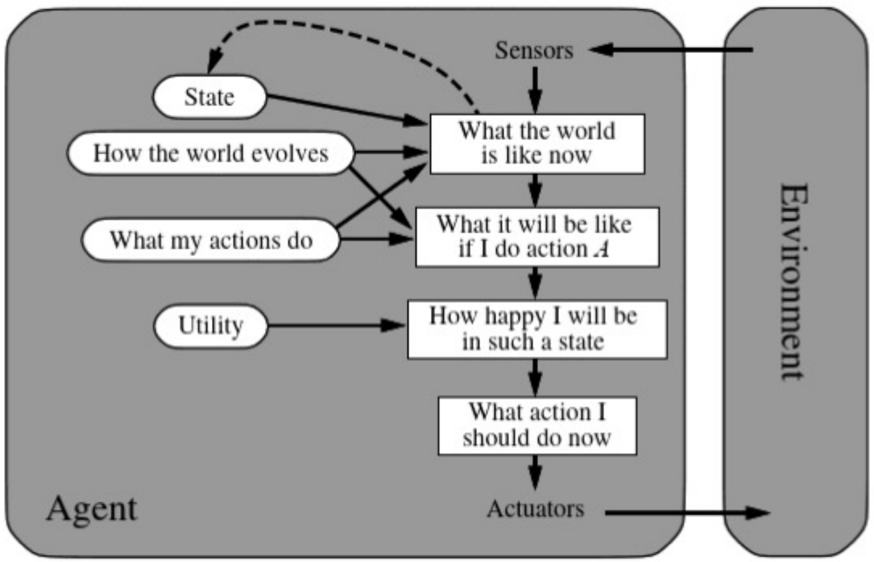


Figure 2.14 A model-based, utility-based agent. It uses a model of the world, along with a utility function that measures its preferences among states of the world. Then it chooses the action that leads to the best expected utility, where expected utility is computed by averaging over all possible outcome states, weighted by the probability of the outcome.

Russel & Norvik. Artificial Intelligence: A Modern Approach (2017),

Autonomous Software Components

ESSAI July 2025

Autonomous Robot Software Components

World Modelling

- Localisation
- Mapping

Navigation

- Path Planning
- Execution and Search
- Obstacle Avoidance

Behaviour Decision Making Planning

Sensor Processing

- Vision
- Audio

Control

- Locomotion
- Grasping
- Motion Planning

ROSBot Maze

World Modelling
- 2D SLAM (Localisation & Mapping)
- Hazard Marker Record
- Traversed Path Tracking
- Local & Global Modelling

Navigation
- Unknown Space Exploration
- Object Search

Search + Recognition

Sensor Processing
- Hazard Marker Recognition

Control
- Differential (or Omni-directional) Drive

RoboCup Soccer

World Modelling
- Localisation
- Ball / Robot persistence
- Team Coordination & Shared State
- Moving object trajectory prediction

Navigation
- Path Planning
- Dynamic obstacle avoidance
- Prediction of kicks

Soccer Skills

Sensor Processing
- Ball Detection
- Field Feature Detection
- Robot Detection

Control
- Bi-pedal Walk & Balance
- Head & Arm Control
- Kicks
- Get-ups

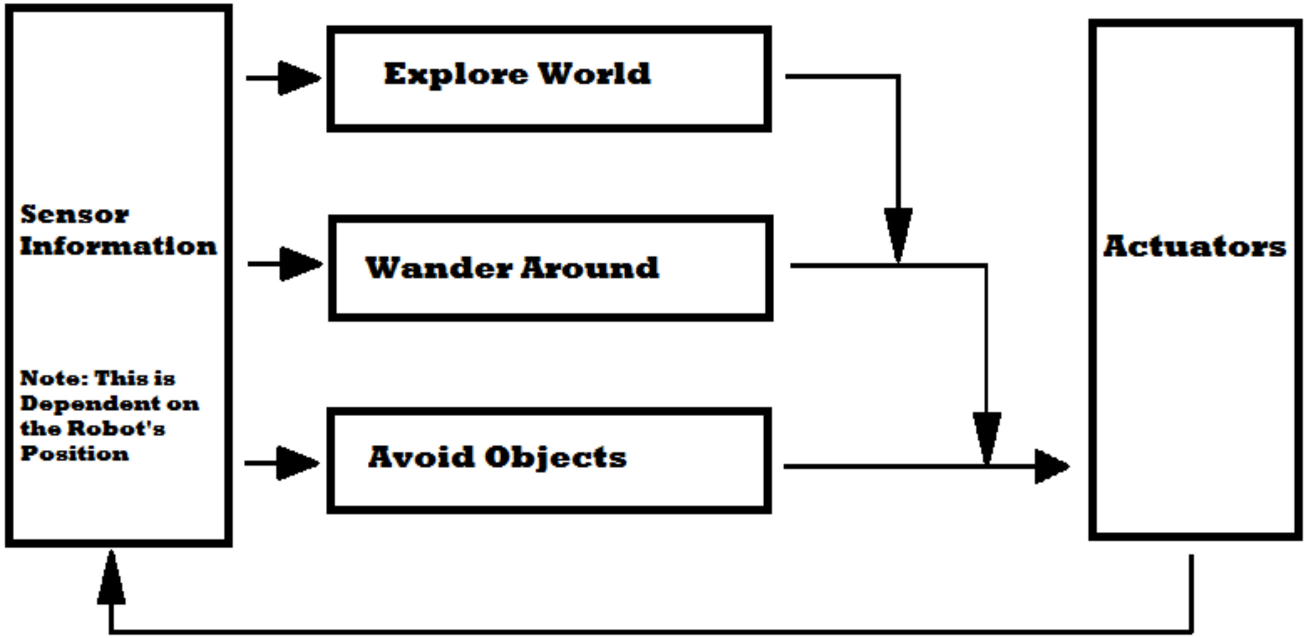
Software Architectures

ESSAI July 2025

Software Architectures: Sense-Plan-Act

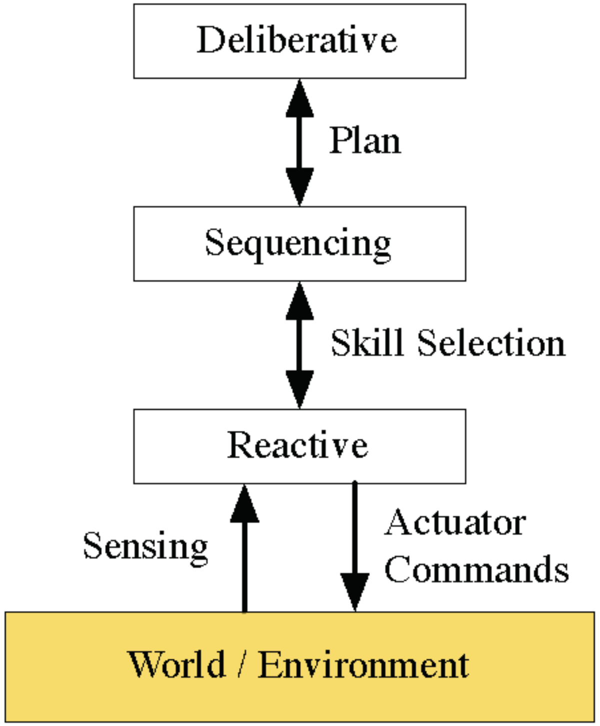
Nilsson, N. J. (1982). Principles of Artificial Intelligence. Berlin Heidelberg: Springer-Verlag

Software Architectures: Subsumption



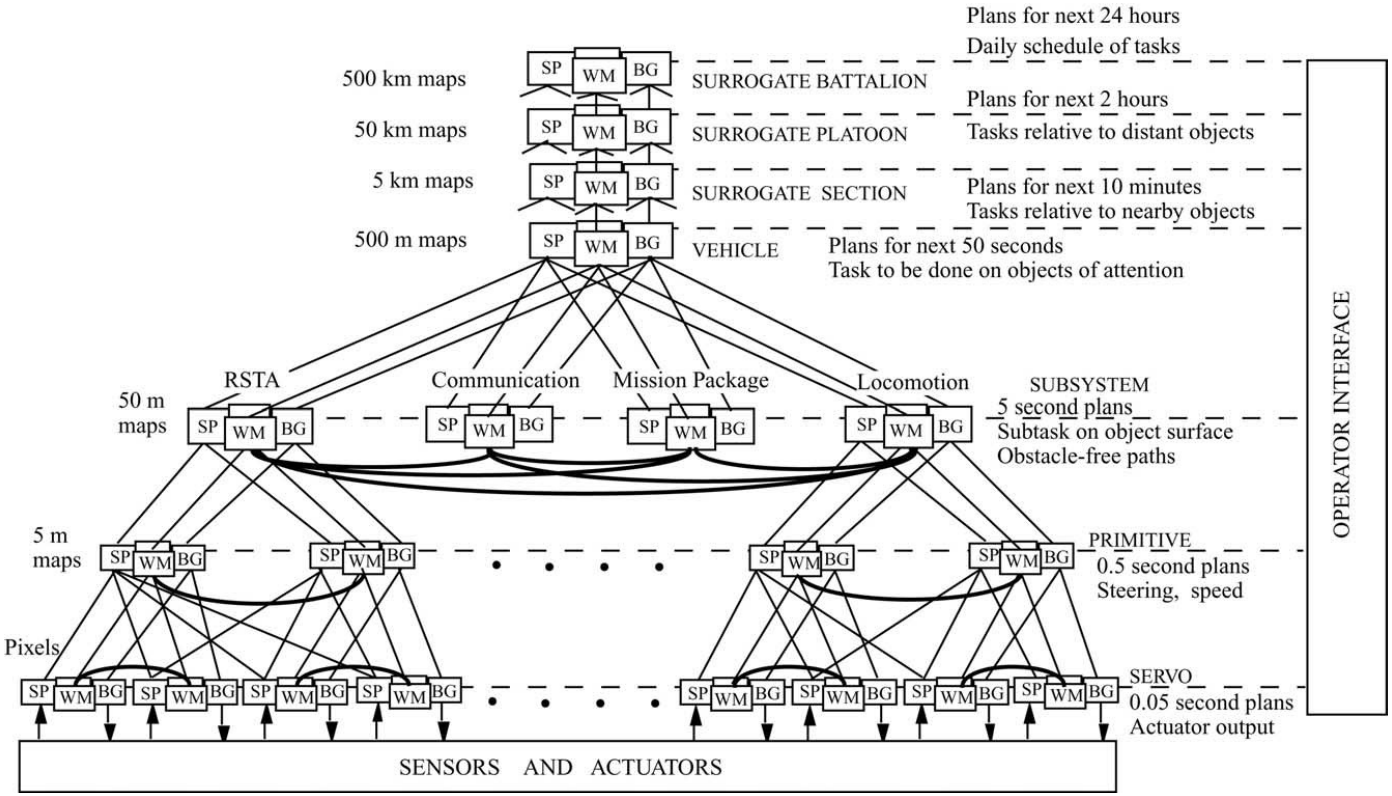
R. A. Brooks (1986), "A Robust Layer Control System for a Mobile Robot", IEEE Journal of Robotics and Automation RA-2, 14-23

Software Architectures: Three-Layer



Bonasso, P. et. al. (1997). Experiences with an architecture for intelligent, reactive agents. Journal of Experimental & Theoretical Artificial Intelligence 9(2-3):237– 256

Software Architectures: RCS



Albus, J. S. & Barbera, A. J. (2005) RCS: A cognitive architecture for intelligent multi-agent systems. Annual Rev Control 29, 87–99.

World Modelling

- Localisation
- Mapping

Navigation

- Path Planning
- Execution and Search
- Obstacle Avoidance

Behaviour Decision Making Planning

Sensor Processing

- Vision
- Audio

Control

- Locomotion
- Grasping
- Motion Planning

DeepMind Learning Soccer

Learning Motion

A practical perspective for Reinforcement Learning

ESSAI July 2025

Day Dreamer

Learning Motion

Algorithms for robot motion must handle a variety of issues unique to the realities of real-world control, including:

- Noise
- Slip
- Hardware failures
- Probabilistic Actions
- Non-deterministic Actions

Reinforcement Learning is a very popular approach as various RL forms can help account for these issues without ‘manual’ intervention.

We could devote a whole course to this topic. For today we’ll keep this to a high-level of how RL methods are applied for robot control, and leave a deep-dive for your own investigations.

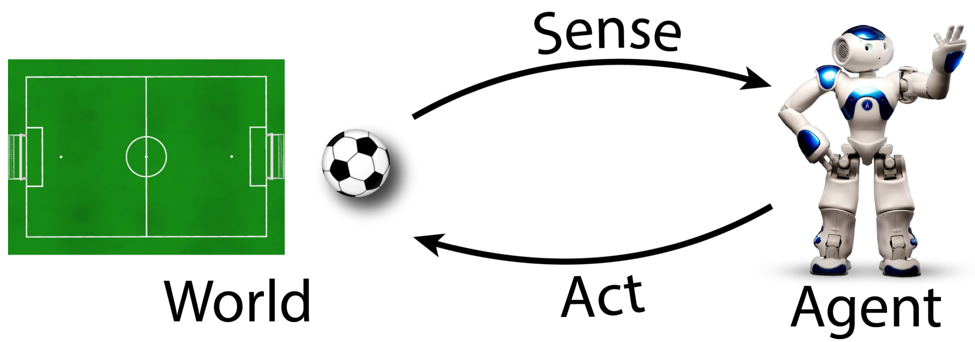
Reinforcement Learning – Overview

An RL problem can be minimally defined as:

$$RL \coloneqq < S, A, T, V, R, \gamma, \pi >$$

Where:

- S – State Space (discretised, continuous)
- A – Action Set (discrete, continuous)
- T – Transition Function (model-based, or model-free)
- V – Value Function
- R – Reward Function
- γ – Discount Factor
- π – Policy Function

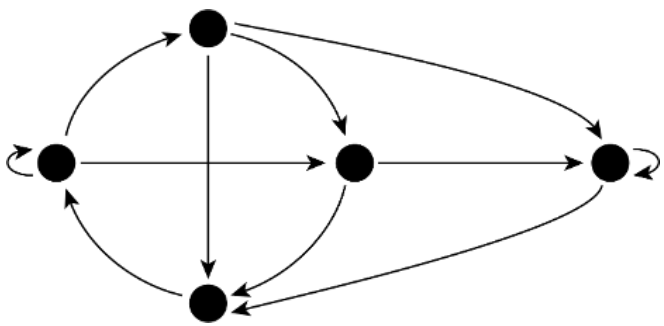


Markov Decision Process

An RL problem defines a Markov Decision Process (MDP) where:

- The decision of which action to perform in a state only requires information that is available in the current state
- The history of states and decisions that the agent took in reaching its current state has no impact on the decision

It is critical that a RL problem is properly defined as an MDP.



Value, Policy, and Q Functions

1. Value function: $V\colon S \rightarrow \mathbb{R}$

$$V(s) = E[r_i + r_{i+1} + \cdots + r_{i+t}] = E\left[r_i \sum_{j=1}^{t} + r_{i+j}\right]$$

$$V(s) = (1-\alpha)V(s) + \alpha \max_a \left[R(s,a) + \gamma V\big(T(s,a)\big)\right]$$

2. Policy function: $\pi\colon S \rightarrow A$

$$\pi^*(s) = \underset{a}{\operatorname{argmax}}[R(s,a) + \gamma V\big(T(s,a)\big)]$$

3. Q-Learning: $Q(S \times A) \rightarrow A$

$$Q(s,a) = (1-\alpha)Q(s,a) + \alpha\left[R(s,a) + \gamma \max_{a'} Q(s',a')\right]$$

Learning Motion

All variates of RL forms can be used within a robotics context:

- On policy learning
  - SARS, Value/Policy iteration
- Off-policy learning
  - Classic Q-learning
- Model-free methods
  - Deep Q-Learning
  - Actor-Critic
  - PPO
- Model-based
  - Dreamer / Day Dreamer

… to name a few

Learning Motion: Issues

Reinforcement Learning methods encounter issues:

- Iterations required for convergence
- Balancing Exploration vs Exploitation
- Appropriate definition of the RL problem
- Transferring simulated behaviours to real-systems

This field is rapidly changing.

Classic Example – Cart and Pole

For simplified domains, a traditional representation with any combination of value iteration, policy iteration, q-learning, etc., are successful:

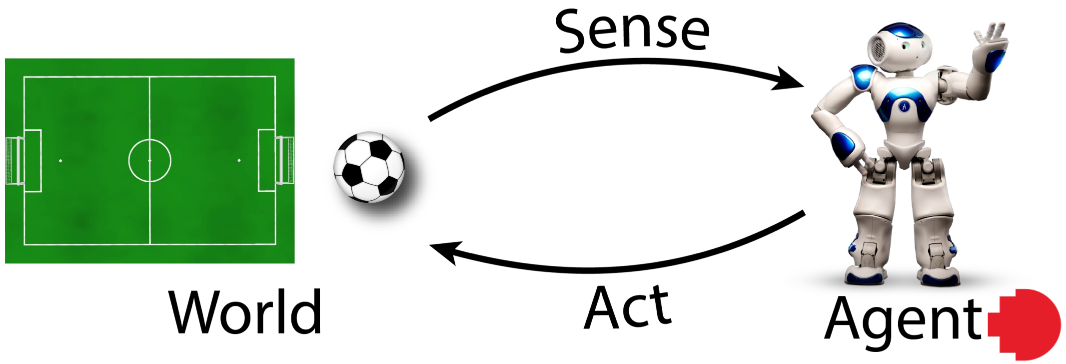
$$RL \coloneqq < S, A, T, V, R, \gamma, \pi >$$

S – Cart/pole position

A – L/R force

T – Either model-based or model-free

R – Balancing time, distance from centre, etc.



Learning Motion: Examples

Deep Reinforcement Learning

Deep Reinforcement Learning methods use the non- linear piecewise function representative power of deep neural networks to encode various parts of the RL problem definition, replacing them with Deep networks.

One of the first successful forms of DRL (2013), is the Deep Q-Network (DQN). Google DeepMind trained a

Model-Free methods: DQN, PPO, SAC, TD3

Model-Based methods: Plan2Explore, Dreamer, Day Dreamer

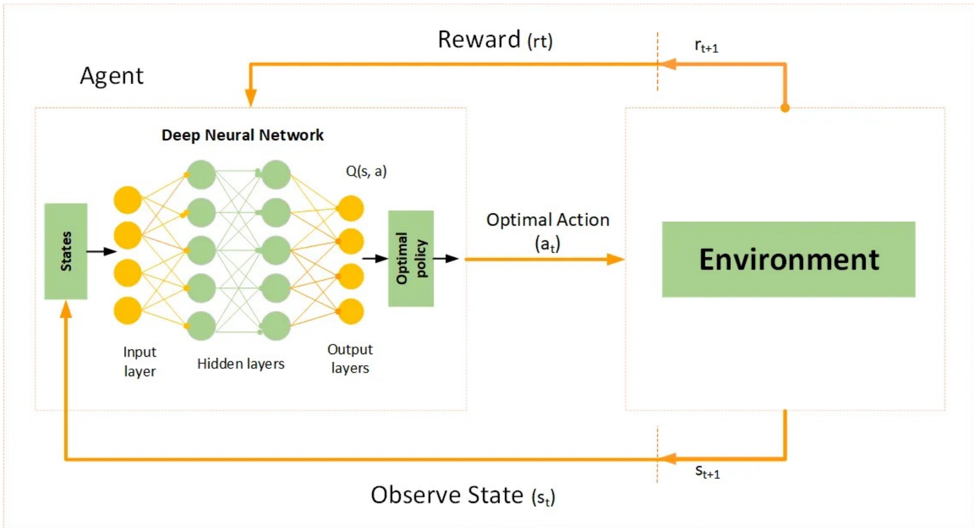
Mnih, Volodymyr, et al. "Playing atari with deep reinforcement learning." arXiv preprint arXiv:1312.5602 (2013).

DQN (Deep Q-Network)

$$RL := < S, A, T, V, R, \gamma, \pi >$$

This essentially represents the Q-learning Q-function $Q(s, a)$ as a Deep Network. In DQN, the Q- function to gives the Loss Function:

$$L(\theta) = \mathrm{E}[y - Q(s, a, \theta)^2]$$



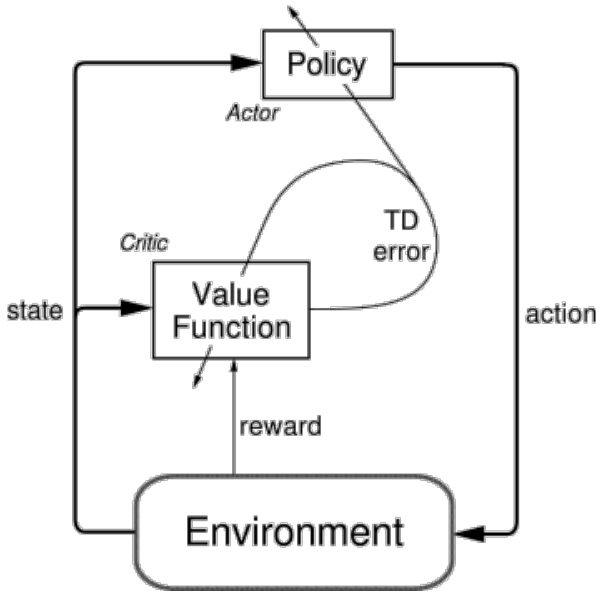
Actor-Critic RL

$$RL := < S, A, T, V, R, \gamma, \pi >$$

Actor-Critic RL, such as TD3 and SAC, out two components:

- **Actor:** decides which action to take in a state, represented by a DNN that outputs action probabilities or deterministic actions
- **Critic:** evaluates the quality of the actions taken by the actor, estimating the value of states or actions (ie. the value function)

This essentially models classic RL, where the value and policy functions are independently computed. The error from the critic updates the actor.



DRL – PPO

PPO (2017) is an extension to SAC and DQN that address the problem of instability in the value/policy networks as well as improving the capability of learning. PPO has now become the standard method for DRL.

PPO introduced concepts of:

- The Advantage
- Clipping
- An Objective Function (to replace the Q- function)

Schulman, John; et. a. (2017), Proximal Policy Optimization Algorithms, arXiv:1707.06347

DRL – PPO

The PPO loss-function is

$$L(\theta) = \mathbb{E}_t\left[\min\left(r_t(\theta)A_t, \mathrm{clip}(r_t(\theta),1-\epsilon, 1+\epsilon)A_t\right)\right]$$

This finds the expectation of the best advantage at each timestep t

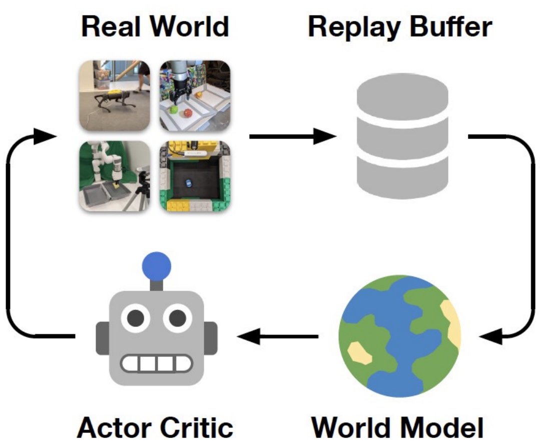
- r_t is the ratio of the probability of selecting an action under the old or new polices, and is not to be confused with the reward!
- A_t is the advantage of an action at timestep t
- ϵ is a hyper-parameter controlling how much each update is clipped

DRL – Dreamer/DayDreamer

As model-free methods don't have an explicit transition function, the the value and policy functions must also “internally encode” approximations of the transitions.

Model-based methods provide an explicit model, which can improve learning. DayDreamer (2023*) is one of the state-of-the-art methods with two deep networks:

- Latent world model models environment dynamics to predict future observations, rewards, and values in latent space.
- Behaviour model - Actor-Critic RL that incorporates world model state predictions



Wu, P., Escontrela, A., Hafner, D., Goldberg, K. & Abbeel, P. DayDreamer: World Models for Physical Robot Learning. in Conference on robot learning 2226–2240 (PMLR, 2023)

Noon Gudgin

Thank you

Day 2:

Motion Planning and Navigation

ESSAI July 2025