# FUNDAMENTALS OF ARTIFICIAL INTELLIGENCE

## Lecture 1: Introduction

**Ola Ringdahl**

UMEÅ UNIVERSITY

# TEACHERS

Ola Ringdahl
ringdahl@cs.umu.se
**Course responsible**

Lois Vanhee,
loisv@cs.umu.se

Andrea Aler Tubella,
aler@cs.umu.se

Adam Dahlgren Lindström
dali@cs.umu.se

Timotheus Kampik,
tkampik@cs.umu.se

Lennart Steinvall,
svartvit@cs.umu.se

Carl-Anton Anserud,
anserud@cs.umu.se

Mail to all teachers on the course:
5dv121ht20-handl@cs.umu.se

UMEÅ UNIVERSITY

# EXPECTED LEARNING OUTCOMES

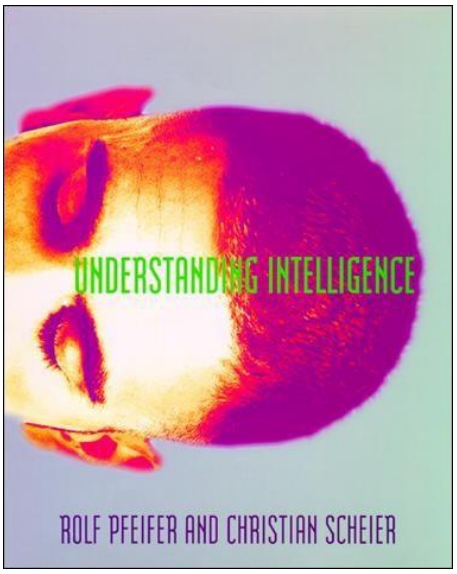
After having completed the course the student will be able to:

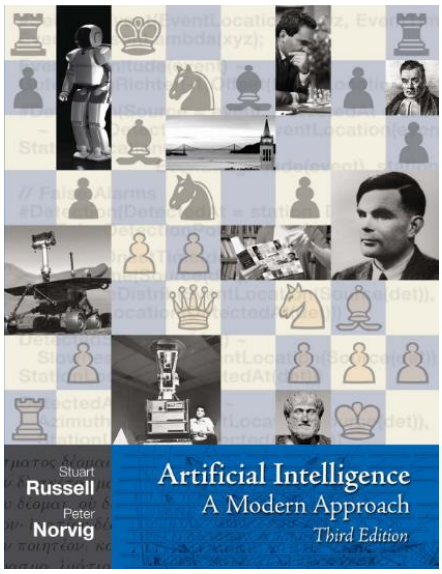
- Give an **overview** of the field **of artificial intelligence**, its background, history, fundamental issues, challenges and main directions
- Interpret and formulate **knowledge representations** in the form of **logic** expressions
- Explain basic concepts, methods and theories for **search**
- Account for **classical planning** of proactive agents
- Describe methods and theories for **reactive agents**, architectures based on subsumption, and potential fields
- Describe the **physical structure of robots**
- Account for different **degrees of autonomy of robots**

# CONTINUED …

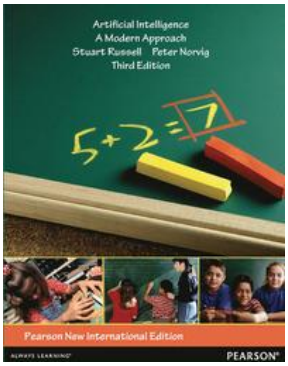
- Explain concepts, methods and theories of **embodied cognition and situatedness**
- Explain basic concepts, methods and theories of
  - Sensing
  - Neural networks and learning
  - Artificial evolution, genetic algorithms, multiple autonomous agents and swarm intelligence
- Demonstrate the ability to **apply a given subset** of the theories, methods and principles discussed during the course
- Discuss and analyze **social implications of AI** technologies in human societies.

# LITERATURE




Plus additional articles/compendia (Cambro).




**NOTE!** *Pearson New International Edition* of the AI book has slightly different chapter numbering. Chapter 7 and 6 have switched places ("Logical Agents“).

# LECTURES

- **Lectures will not cover all content of the course,** but will introduce each topic of the course
  - I'm sorry but you also have to read the books ☺
- A set of readings is given for each lecture (together with the slides) on Canvas
- Some lectures will be **flipped calssroom**
- You are always welcome to ask questions during the lectures
- You are encouraged to send questions or topics via mail, we will bring them up during the lecture

# EXAM AND GRADING

- The course is graded using in the following way:
  - U – Fail
  - 3 – Pass
  - 4 – Pass with mark
  - 5 – Pass with distinction
- In order to receive a grade of 3 or higher you must:
  - Pass the exam
  - Complete all obligatory assignments
  - Attend to obligatory parts of the course
- Assignments are only graded with Pass/Fail

# COURSE EVALUATION

- Result of last years student evaluation (68/163 respondents):

| Good things | Could be improved |
|---|---|
| The assignments and seminars | Some assignments needs to be improved |
| Good contents of the course | Short time for ANN assignment. Too long for follow the path |
| Assignment assistance | Schedule assistance hours earlier |

# CHANGES TO THIS YEAR

- Everything online due to Covid
  - Note that access to computer labs are restricted. Use your own computer if possible
- Removed one of the seminars and changed the ethics seminar somewhat
- More time for ANN assignment, less for follow the path
- Many lectures reworked
- Most tutoring hours scheduled already now
  - Check the Canvas schedule

# ASSIGNMENTS

**There are 4 assignments**
**(3 for BSc cognitive science students, 5DV201)**

UMEÅ UNIVERSITY

# ASSIGNMENT 1: FOLLOW THE PATH

- Create a program that controls the robot in a simulated environment such that the robot follows a specified path
- Work in pairs, **demonstrate** your working solution **on Zoom**
- After demonstration, **compare** your results to another group and write a **1 page report** about similarities/differences together
- Demo: 2020-09-16
- Report: 2020-09-17

# INTRODUCTION TO MRDS

- There is an online **video introduction** to the virtual robot environment used for the first assignment, MRDS available on Canvas
- The goal is that you should be able to start the environment and move the robot around using the given code
- Make sure to install MRDS and get the demo code working already now!

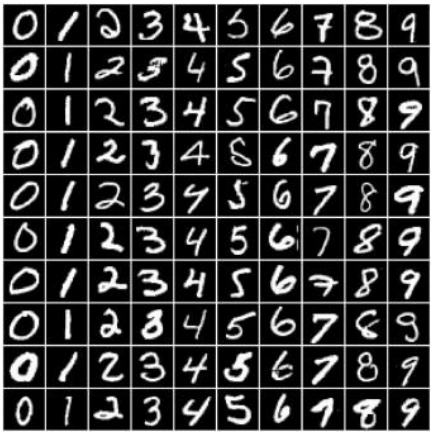
# ASSIGNMENT 2: K-NN

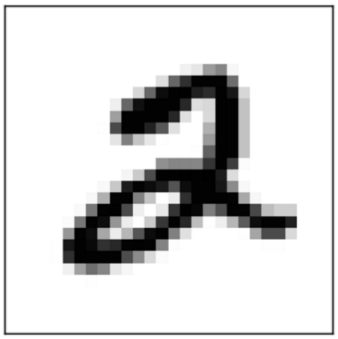
- Investigate how the **k-nearest neighbor-algorithm** works
- Work together in groups of two
- There will be an online **workshop** where we will work on it together (Sept. 21)
- Deadline: 2020-09-28

- Note: not for BSc cognitive science students (5DV201)

# ASSIGNMENT 3: NEURAL NETWORK

- Create a simple Artificial Neural Network (perceptron)
- Train the perceptron such that it can classify hand written numbers (from MNIST database)
- BSc cognitive science (5DV201) have a simpler assignment using a pre-implemented network recognizing faces
- Deadline 2020-10-14





UMEÅ UNIVERSITY

# ASSIGNMENT 4: ETHICS OF AI – ROLE-PLAY SESSION

- Scenario: A Facebook-like website wants to revise which news and ads are shown to users
- Task:
  - You **will play the** characters that represent **stakeholder** groups (3 ppl) involved in this transition during the role-play session:
    - users, company representatives, governments
  - **Negotiate with each other** how to revise the news and ads algorithm
  - After the role-play you **hand in a report**, which partly depends on the negotiation.
- Deadlines:
  - Stakeholder group proposal: 16th of October
  - Negotiation (role-play): 19th of October
  - Final report: 21st of October

# TIGHT SCHEDULE!!

- Very tight deadlines for the assignments
- Make sure to follow the planning on Canvas
- Make sure to check the schedule often as it might change

# CHEATING AND PLAGIARISM

- Read this document so that you are familiar with the rules about cheating!
- **You are not allowed to copy** code or text from other students or the internet.
  - It is allowed to discuss your solutions as long as you don't copy from each other
- Be sure to reference any information that you used
- The university takes this seriously and several students are warned or suspended every year

# TUTORING

- You can get help through the **discussion forum on Canvas**
  - Write there for help (or check previous answers for tips)
  - There is a also thread for finding a partner for the assignments
- We will have scheduled **tutoring through Zoom**
  - On Tutoring Queue (https://webapps.cs.umu.se/tutorqueue) write your zoom link and a tutor will connect when available
  - Look at the schedule on Canvas (normally Mon + Thu 10-12)
- Mail: 5dv124ht20-handl@cs.umu.se (reaches all teachers on the course)
  - Better to use discussion forum and tutoring hours if possible.

# INTRODUCTION TO AI

**Readings:**
**Russel and Norvig Ch. 1**

**Pfeifer and Scheier Ch. 1**

UMEÅ UNIVERSITY

# ARTIFICIAL INTELLIGENCE

| | |
|---|---|
| **Thinking Humanly**<br>Cognitive science approach. Try to model how humans actually thinks. *General Problem Solver (GPS)* | **Thinking Rationally**<br>"Think" using logic. Rule based |
| **Acting Humanly**<br>Passing the *Turing test*. Could pass as a human<br><br>"make computers do things that humans, at the moment, are better at" | **Acting Rationally**<br>Rational agents: always act to achieve the best outcome. |

Russell & Norvig (2010), p. 2

# THE FIELDS OF AI

Philosophy

Computing Science

Psychology

Neuroscience

Linguistics

Mathematics

Economics

Control theory

UMEÅ UNIVERSITY

# WHAT DO WE STUDY

- **Intelligence**
- Thought
- **Reasoning**
- **Rationality**
- **Learning**
- **Memory**
- **Representation**
- Consciousness
- Attention
- Creativity
- **Behavior**
- **Emergence**
- **Robotics**
- **Sensing**

# THE BIRTH OF AI

- The Dartmouth Conference, 1956
  - John McCharthy
  - Marvin Minsky
  - Allen Newell
  - Herbert Simon
- Symposium of Information Theory, MIT, 1956
  - Allen Newell (computer science)
  - Herbert Simon (political science)
  - George Miller (psychology)
  - Noam Chomsky (linguistics)
  - Jerome Bruner, Jacqueline Goodnow and George Austin (psychology)
- Also the birth of Cognitive Science as we know it

# DEFINING INTELLIGENCE

1. Ability to carry out abstract thinking
2. Ability to think rationally
3. Ability to act rationally
4. Playing chess
5. Scoring high at an IQ-test
6. Knowing a lot of stuff
7. Ability to learn and adjust oneself to the environment
8. A biological mechanism by which the effects of a complexity of stimuli are brought together and given a somewhat unified effect in behavior
9. The capacity to learn or profit from experience

Given some behavior – say of a human, an elephant, an ant, or a robot – that we find interesting in some way, how does this behavior come about?

(Phfeifer and Scheier, 2001)

# DIVERSITY-COMPLIANCE TRADE-OFF

Almost all notions of intelligence introduce a trade-off between *doing what you know* and *doing something new*. This tradeoff has many names and we will run into several of them during this course.

- **Psychology:** Assimilation – Accommodation
- **Adaptivity:** Conservative – Innovative
- **Learning theory:** Flexibility – Stability
- **Reinforcement:** Exploration – Exploitation
- **Evolution:** Mutation – Inheritance

# HOW TO STUDY INTELLIGENCE

## Reductionism

- Break down the problem into manageable pieces
- Study each part separately
- Put them together and understand the whole
- Typically a *top-down* perspective

## Holism

- Argues that we can not break down the system into parts without loosing important properties of the system that does not exists in any of the parts alone
  - These properties are called *emergent*
- Studies simple but complete systems
- Typically a *bottom-up* perspective

# ANALYTIC APPROACH

- Typically reductionistic
  - Break the system of analysis into parts and analyze each part
- Primary methods of all empirical sciences

- A hypothesis of how the system works is created
- Experiments are designed to test the hypothesis
- Experiments are performed and data is gathered
- Results are analyzed

# THE ANALYTIC APPROACH IN AI

## Example: Explain how a car works

1. Start by observing the behavior of the car
    1. Measure how quickly it accelerates, turns, breaks, etc…
2. Try to come up with an explanation for the observed behavior
    1. Based on the explanation, come up with a number of testable hypotheses
3. Design an experiment that can falsify the hypotheses
4. Run the experiments and analyze the results

The car can of course be replaced with a human, animal etc…

# THE SYNTHETIC APPROACH

- *Understanding by building*
- Modeling
  - Build a model of the biological system
    - The model should replicate some aspects of the natural system
  - Use the model in order to predict the results of traditional experiments
  - Use the model to perform experiments that can not be performed on the natural system

- AI and Cognitive Science is to the large part synthetic
- The synthetic and analytic approaches are complementary, not contradictory!

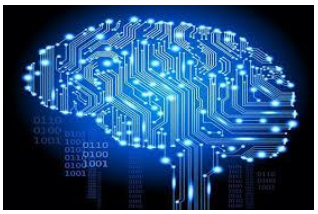
# THE SYNTHETIC APPROACH IN AI

Example: Explain how a car works

1. Start by observing the construction of the car
    1. Study how the engine works, how the suspension and wheels are constructed…
2. Try to come up with a model that matches observations
3. Implement the model
    1. Physical model (robot) or simulated in a computer
4. Run the model and compare the results with the observed behavior of the car
5. Make predictions of which results we should expect if we do some new experiment using a real car

The car can of course be replaced with a human, animal etc…

# INFORMATION PROCESSING VIEW



- The brain is viewed as a computer
  - The brain receives input from our senses
  - The inputs are processed, producing mental representations
  - Based on the internal mental state, an intention or plan is generated
  - Finally, actions are executed, producing an output
- Also known as **Sense – Think – Act**
- This is a metaphor, we do not mean that the brain internally works as a computer
- This view is important for cognitive psychology and cognitive science in general

# HISTORY OF AI

## Examples of applications

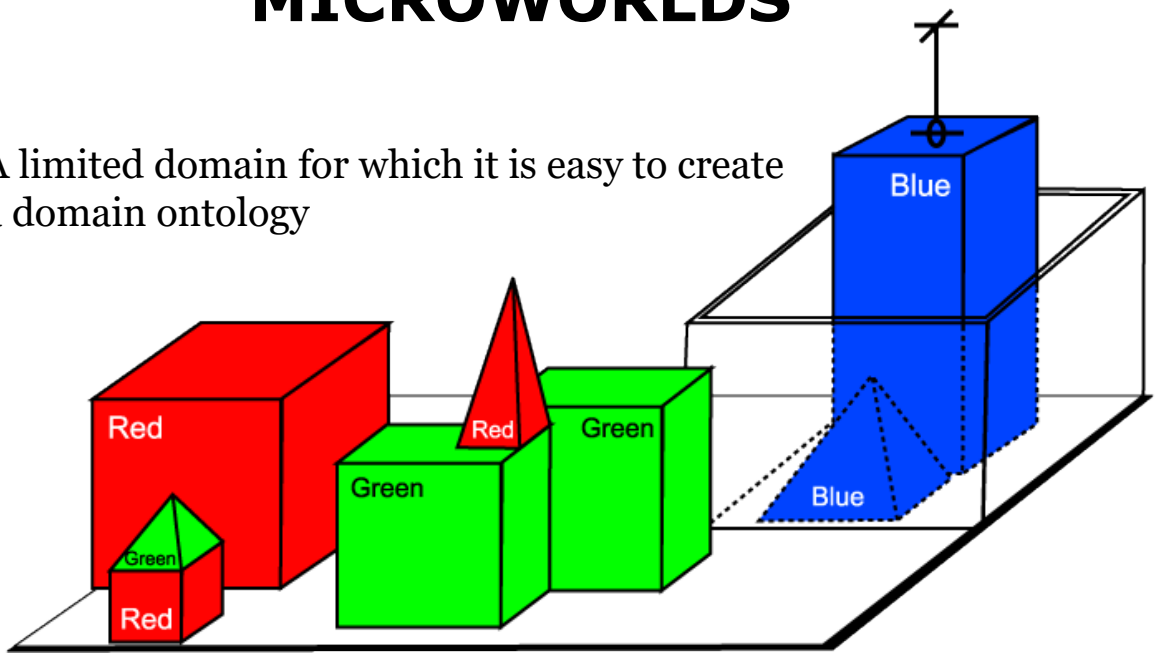
UMEÅ UNIVERSITY

# GENERAL PROBLEM SOLVER (GPS)

- Presented by Herbert Simon and Allan Newell in 1957
- In principle, GPS can solve any formalized problem
- The problem is specified as a set of rules
- GPS will search for a solution
  - GPS searches using a heuristic called Means-Ends Analysis
- Used to prove theorems, geometric problems, chess problems, and similar
- Could not solve many real world problems due to combinatorial explosion
  - I.e., the program worked in principle but required enormous time to solve large problems

# MICROWORLDS

- A limited domain for which it is easy to create a domain ontology




SHRDLU, written by Terry Winograd of MIT, 1972

# EXPERT SYSTEMS

## Example: Mycin (1972)

- Often said to be the first real knowledge based AI system (expert system)
- It was a medical decision support system
- Like the micro-world systems, it performs in a limited domain
- Knowledge is represented as rules:

**RULE543**

**IF:**

1) The infection which requires therapy is meningitis,

2) Only circumstantial evidence is available for this case,

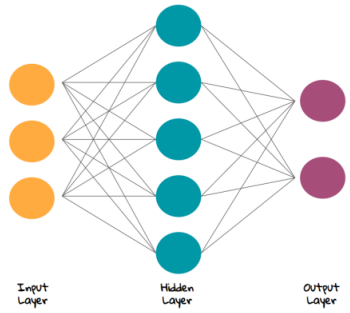
3) The type of the infection is bacterial,

4) The patient is receiving corticosteroids,

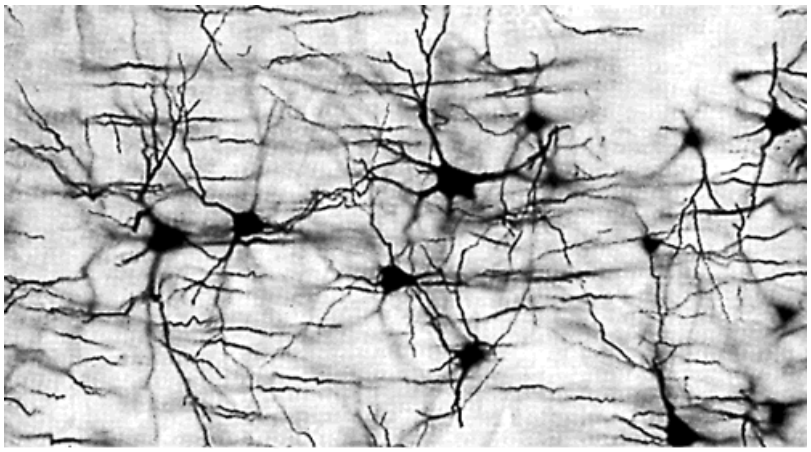
**THEN:**

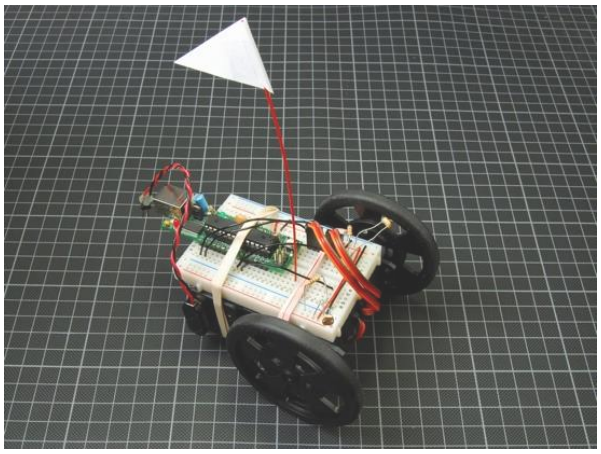
There is evidence that the organisms which might be causing the infection are e.coli (.4), klebsiella-pneumoniae (.2), or pseudomonas-aeruginos

# THE GREAT 80-IES




- The return of Artificial Neural Networks
  - *Birth of connectionism*
- Robots that can act in the real world!
  - *Elephant's don't play chess!* – Brooks (1990)





# THE REALISTIC 90-IES

- AI becomes less of an art and more like a science
- Mathematics and statistics is emphasized
  - Hidden Markov Model (e.g. for speech recognition)
  - Kalman filter (filter out noisy sensor data)
- The complete agent – autonomous, embodied, situated
  - Need too tie isolated subfields together
  - Reasoning systems must handle uncertainty
- Artificial General Intelligence (AIG)
  - Attempt to construct human-like AI
  - Reaction against microworlds and domain-specific approaches

# 2000 UNTIL TODAY

- AI becomes an industry
  - AI technologies are applied for everyday use
- Scale it up!
  - Really large datasets
- “Super computers” (cloud computing / clusters)
- Robotics
- Mobile technologies
  - More sensors
- Deep Learning !

# STATE OF THE ART

- When it starts working we stop calling it AI
  - Unlocking the phone using face recognition
  - Autopilot or airplanes
  - GPS navigator / Google Maps telling us how to drive
  - Spam detection algorithms
  - Netflix recommendation algorithms
  - Automatic recognition of what's in photos (e.g. search for a picture of a cat on a beach)
- Big datasets needed!
- Google has released an AI Hub
  - Use (/buy) ready made/trained models

# STATE OF THE ART

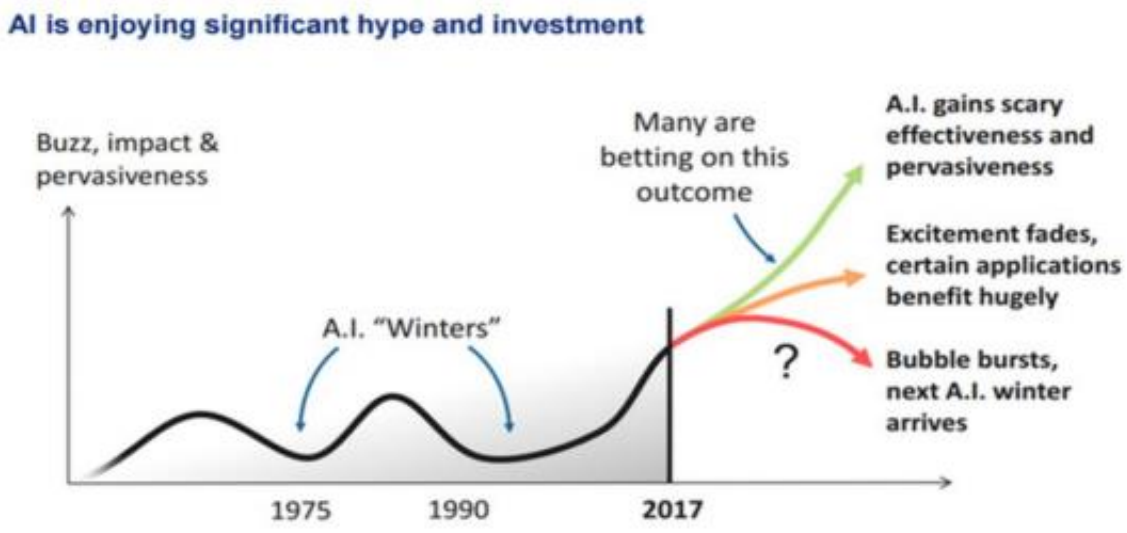
- Machine learning
- Natural language processing (NLP)
  - Machine translation
  - Speech recognition
  - Chat bots
- Robotics
  - Autonomous cars
  - Robot vacuum cleaners / lawn mowers
  - Agriculture robots (harvesting/weeding/plowing/...)
- Game playing
- Generating fake videos

# MAIN CHALLENGES

- Explainability – Technical challenge
- Algorithmic bias – Technical/societal challenge
- Transparency in usage - Societal/political/legal challenge
- Applications - Societal/political/legal challenge

UMEÅ UNIVERSITY

# FUTURE OF AI?


AI is enjoying significant hype and investment
Buzz, impact & pervasiveness
Many are betting on this outcome
A.I. gains scary effectiveness and pervasiveness
Excitement fades, certain applications benefit hugely
Bubble bursts, next A.I. winter arrives
A.I. "Winters"
?
1975
1990
2017

# CURRENT MAJOR EFFORTS IN AI

- WASP - Wallenberg AI, Autonomous Systems and Software Program
  - Sweden's largest individual research program ever
    - Many researchers @umu involved
  - Vision: **excellent research and competence in AI**,
- AI4EU - seeks to develop a **European AI ecosystem**,
  - Establish an **Ethics** Observatory - Virginia Dignum (cs@umu)
  - Create an open **AI platform**
- AI Competence for Sweden – **increasing AI competence in Sweden** by developing courses for students and industry
  - UMU one of 7 universities participating
- AI@UMU: https://www.umu.se/en/research/features-and-news/artificial-intelligence/

# THINGS TO DO AS SOON AS POSSIBLE !

- **Form groups** (of two) for Assignment 1
  - See the assignment specification on the course web for more information
  - Use the discussion forum on Canvas to find a partner if needed
- Install MRDS on your computer and make sure the demo code works for you
  - Look at the instruction video on Canvas
- Remember to **register for the course** (through the student web)

# INTRODUCTION TO ROBOTICS, PATH TRACKING, ROBOT ANATOMY

**Ola Ringdahl**

UMEÅ UNIVERSITY

# ROBOTICS LECTURES IN THIS COURSE

L1
- Introduction to robotics
- Path tracking

L2
- Robot anatomy
- Potential fields

L3
- Robot paradigms
- ~~Biological inspiration~~

UMEÅ UNIVERSITY

# CONTENTS OF THIS LECTURE

1. Introduction to robotics
2. Path tracking
3. Robot anatomy (part 1)

# INTRODUCTION TO ROBOTICS

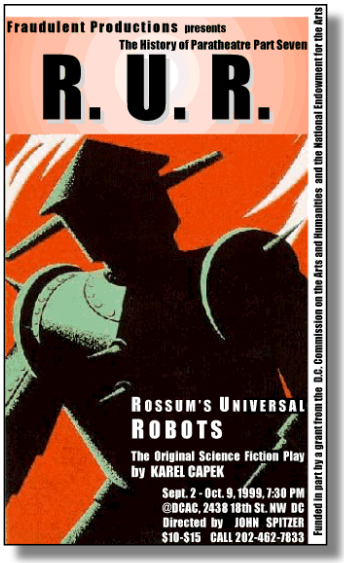
**Readings:**
**Dudek - Overview (p1-11).pdf**

UMEÅ UNIVERSITY

# CONTENTS OF PART 1

- What are we talking about?
- Different kinds of robots
- Why do we need them at all?
- The prehistory of Robotics
- Challenges

# THE WORD "ROBOT"

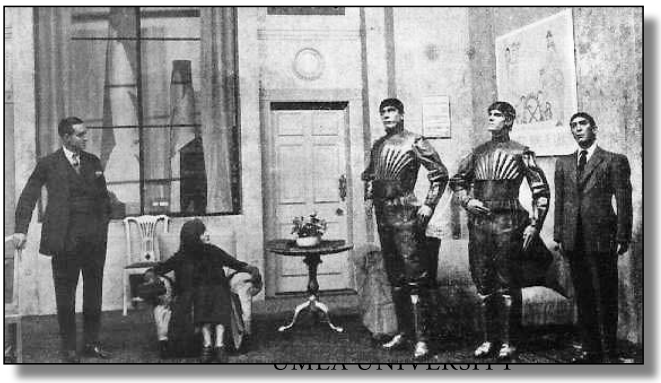


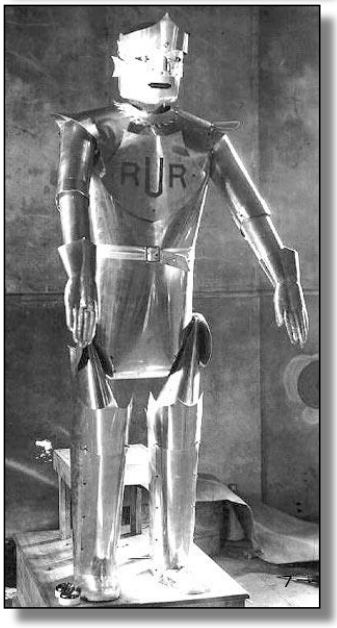

It was first used in 1920 in a play called "R.U.R." or "Rossum's Universal Robots" by the Czech writer Karel Capek.

The story: Man makes robot - then robot kills man!

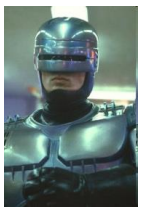
Many movies continued on this theme.

"Robot" also sometimes applies to programs that search the web (bots, spiders)





# CURRENT STATE OF ROBOTICS



- Robots of today are not the walking talking intelligent machines of movies and books.

- Until ~2014, most robots worked in factories, warehouses, and laboratories.

- Since then, most robots are service robots: lawnmowers and vacuum cleaners.

- Now robots are starting to show up in our schools, stores, offices, and even in our bodies.

# DEFINITIONS OF "ROBOT"

- Robot Industry Association (RIA):

"A re-programmable, multifunctional manipulator designed to move material, parts, tools, or specialized devices through variable programmable motions for the performance of a variety of tasks".

- An "AI definition":
  A machine able to extract information from its environment and use knowledge about its world to move safely in a *meaningful* and *purposive* manner.

# TWO MAJOR TYPES OF ROBOTS

**Industrial robots**

- Operate in a stable deterministic environment
- Normally fixed or restricted mobility
- Operate with no or limited interaction with the world
    - Although collaborative robots becomes more common

**Service robots**

- Operate in “the real world”
- Move around
- Require a high degree of autonomy
- Interact with the environment

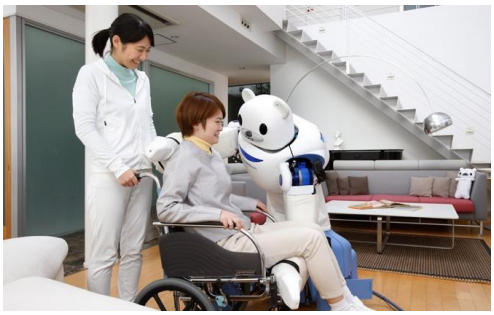
UMEÅ UNIVERSITY

# PROFESSIONAL SERVICE ROBOTS

- **harvesting robots**
- **transportation robots**
  - agriculture, forestry, mining, …
- **inspection and maintenance**
  - pipelines, sewers, pool cleaning
- **medical robots**
  - remote surgeon
- **defense**
  - surveillance, patrolling, shooting, bomb/mine disposal
- **rescue and security robots**
  - USAR, patrolling
- **construction and demolition robots**
- **space robots**
  - exploration, assembly & assistance

# PERSONAL SERVICE ROBOTS

- customer relations
- home security
- handicap/elderly assistance
- education
- domestic



UMEÅ UNIVERSITY

# WHY ROBOTS?

5

The ~~4~~ D's:

- Dirty
- Dull
- Dangerous
- Difficult
- Distant

$ Dollar

UMEÅ UNIVERSITY

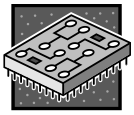
# WHY ROBOTS?
## (SPECIFICALLY FOR INDUSTRIAL ROBOTS)

To

- Improve productivity
- Increase production
- Improve working conditions
- Improve quality
- Improve flexibility
- Lack of workers

# THE PREHISTORY OF ROBOTICS

- Control

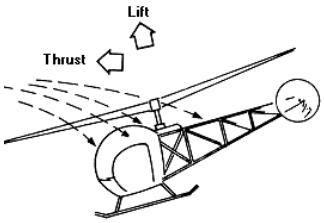


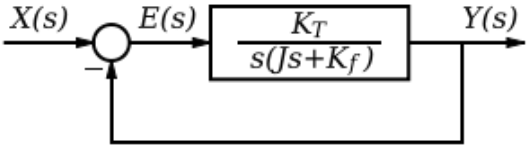
- Cybernetics

- Artificial intelligence

UMEÅ UNIVERSITY

# The prehistory of Robotics… Control



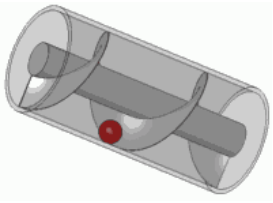



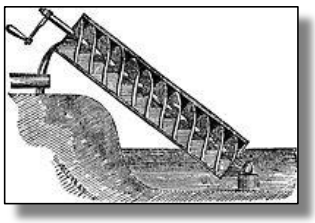

**What is Control?**

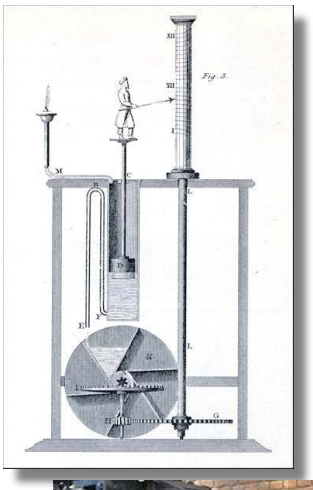
Fundamental concepts governing all mechanical systems (steam engines, airplanes, cranes, etc.)

Thought to have originated with the ancient Greeks:

- Clocks
- Water pumps







Forgotten and rediscovered in Renaissance Europe:

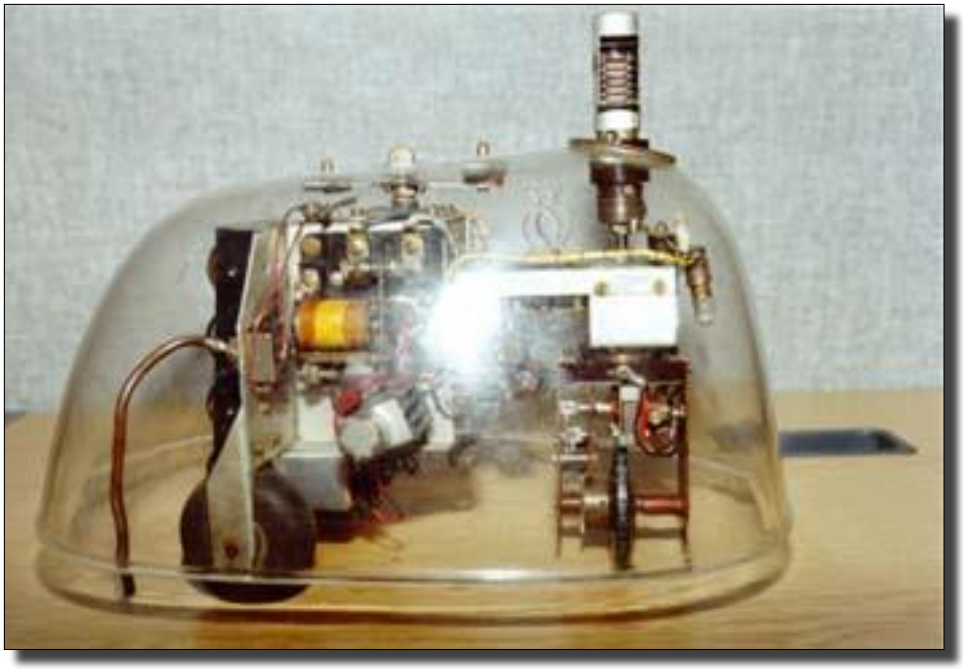
- Heat-regulated furnaces
- Windmills
- James Watt's steam engine (the governor)

# THE PREHISTORY OF ROBOTICS… CYBERNETICS

- A combination of biology, information science, control theory
- Developed by Norbert Wiener in the late 1940
- Seeks to explain the common principles behind control and communication in animals and machines
- Studied the coupling between an organism and its environment

# 1953: GRAY WALTER'S TORTOISE

- Seeking light
- Head toward weak light
- Back away from bright light
- Turn and push
  (for obstacle avoidance)
- Recharge battery



UMEÅ UNIVERSITY

# THE PREHISTORY OF ROBOTICS … AI

AI was born at the Dartmouth Research Conference in August 1955

Marvin Minsky 1955:
"[an intelligent machine] would tend to build up within itself an abstract model of the environment in which it is placed. If it were given a problem it could first explore solutions within the internal abstract model of the environment and then attempt external experiments".

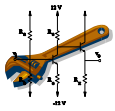
This view dominated AI and robot research for 30 years.

Shakey (1966-72)

# ROBOTICS: MULTIDISCIPLINARY

**Classical AI**

- Knowledge representation, Natural language processing,
- Planning, Searching
- Perception



**Mechatronics**

- Sensors, Actuators

**Control theory**

- To control motion and manipulation

# ROBOTICS: MULTIDISCIPLINARY

**Signal processing**

- Image processing, Prediction, Filtering

**Machine learning**

- Pattern recognition, Neural networks, Fuzzy logic, Genetic algorithms

**Computer Science**

- Software engineering, Architectures

**Numerical methods**

- Control, Optimization, Parameter estimation in models

# ROBOTICS: MULTIDISCIPLINARY

**Neurophysiology**

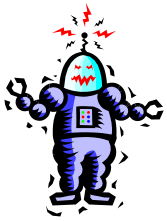
- Human control systems

**Ethology**

- Animal behavior

**Psychology**

- Human behavior



**Robotics**

- Navigation (Localization, Path planning, Map making, Obstacle avoidance),
- Human-Robot Interaction (HRI), Robot ethics

# CHALLENGES

**Perception**

- Unstructured environments (Object recognition and Speech recognition)

**Control**

- Manipulation of objects in the real world
- Limited capabilities of robot actuators/effectors
- Power consumption

**Thinking**

- Situational awareness, Problem solving, Knowledge representation, Natural language processing (NLP), Learning

**System level**

- Robot safety
- Human-robot interaction (HRI)
- Networks of robots

# PATH TRACKING FOR MOBILE ROBOTS

**Readings:**

**Coulter 1992 Pure Pursuit.pdf**

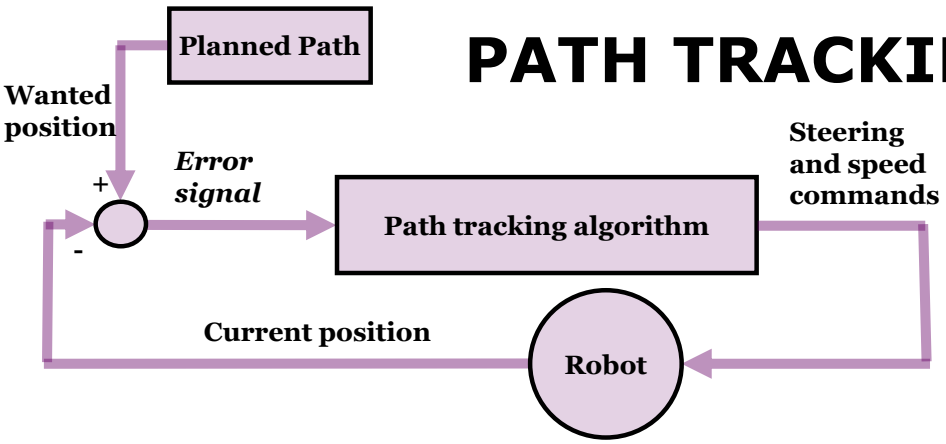
**Ringdahl 2003 Master thesis.pdf (Ch. 3)**

UMEÅ UNIVERSITY

# ROBOT NAVIGATION

- We want to control where the robot goes!
- Often we know where the robot is, and where it should go (but not how)
- Path planning creates a path
  - More in the next AI course
  - and with Potential fields later on in this course
- Sometimes a path is given
  - like in the assignment

# PATH TRACKING



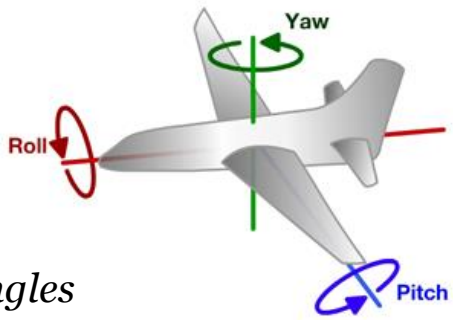

- The control algorithm that makes the robot follow a given path
- Intuitively: If to the left of the path: turn right. If to the right of the path: turn left. But don't turn too little, and not too much!
- In general: A non linear control problem
- Two intuitive algorithms:
  - Follow-the-carrot
  - Pure pursuit

# THE ROBOT POSE

- Describes a robot's location in 3D space
- A mobile robot has 6 degrees of freedom (DOF) expressed by the pose: $(x, y, z, \phi, \theta, \psi)$
- Position = (x,y,z)
- Attitude = (roll, pitch, yaw)

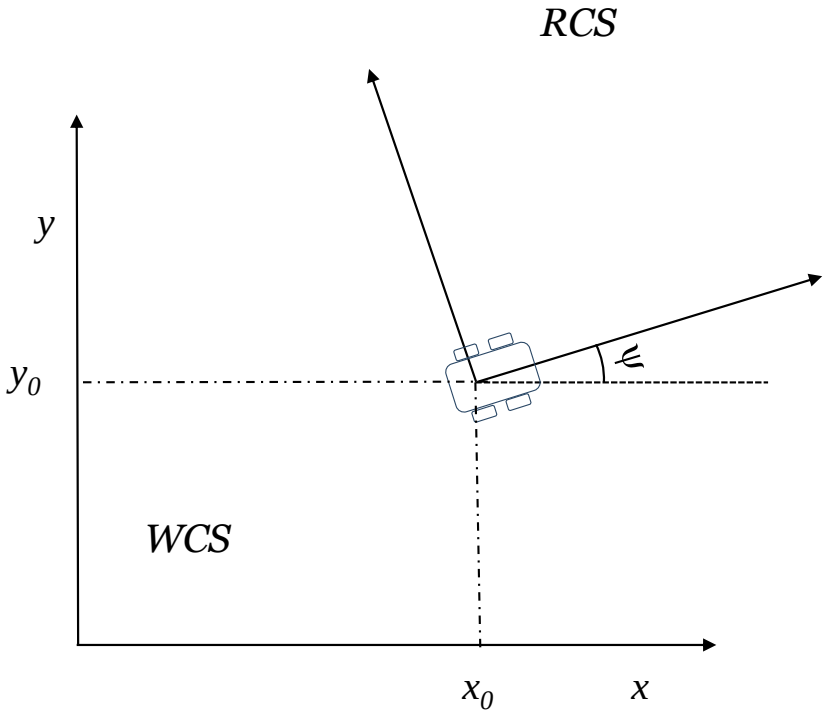
  $= (\phi, \theta, \psi)$

  = (phi, theta, psi)
- roll, pitch, and yaw are denoted *Euler angles*



# TWO 2D COORDINATE SYSTEMS

- The robot has its own coordinate system (RCS)
- So does the world (WCS)
- The robot's pose describes the relation between RCS and WCS
- For indoors robots, we often make the *"flat ground assumption"*: The robot moves in a flat 2D world. The pose is then simply (x, y, ψ) where ψ is the yaw (a.k.a. heading)



# CONVERSION BETWEEN COORDINATE SYSTEMS

- Let $(x_o, y_o, \psi)_{WCS}$ denote the robot's pose
- Let $P = (x', y')_{RCS} = (x^w, y^w)_{WCS}$
- Then:

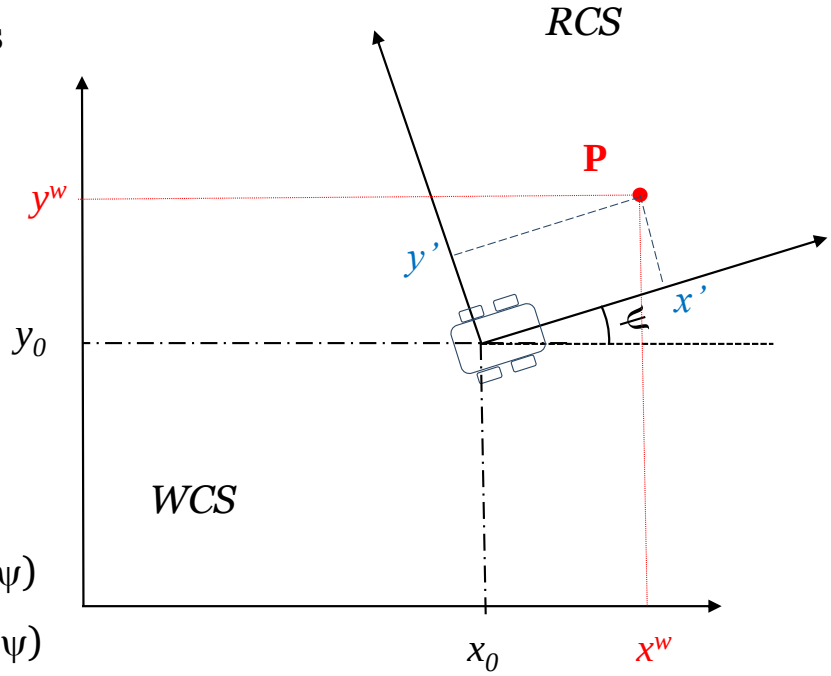
$$x^w = x_o + x' \cos(\psi) - y' \sin(\psi)$$

$$y^w = y_o + x' \sin(\psi) + y' \cos(\psi)$$

- And the other way around:

$$x' = (x^w - x_o) \cos(\psi) + (y^w - y_o) \sin(\psi)$$

$$y' = -(x^w - x_o) \sin(\psi) + (y^w - y_o) \cos(\psi)$$



# EULER ANGLES & QUATERNIONS

- *Euler angles* is a way of specifying the angular relation between two coordinate systems
- A problem with Euler angles: one or several angels may not be uniquely defined (denoted *gimbal lock*)
  - Example: What is the yaw of a vehicle moving straight up (pitch=90 degrees)?
  - For this reason, the space industry adopted a different definition of the angles
- *Quaternions* is another representation, and are
  - not susceptible to gimbal lock
  - more compact, quicker to compute and more numerically stable

# EULER ANGLES & QUATERNIONS

- According to Euler's rotation theorem, any sequence of rotations around several axes is equivalent to a single rotation by a given angle about a fixed axis
- A quaternion represents this axis and the amount of rotation with four numbers, usually denoted (w,x,y,z)
- The MRDS simulator provides quaternions representing the robot's attitude (corresponding to roll, pitch and yaw in Euler)
- You can convert quaternions to Euler angles by calling a subroutine
    - Don't try to make sense of the quaternion, just convert it!

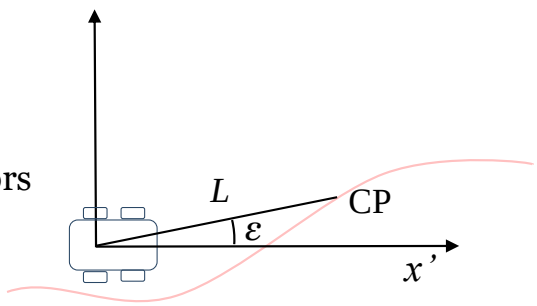
# FOLLOW-THE-CARROT

- A simple and common algorithm for path tracking
- Analogy: a driver sitting in a cart pulled by a donkey steers by dangling a carrot in the desired direction in front of the donkey using a fishing-rod
- Algorithm: At each time step, the robot steers towards a point further along the route, with no concern about the appearance of the route before that point.

Barton, M.J., Controller Development and implementation for Path Planning and Following in an Autonomous Urban Vehicle. University of Sydney, November, 2001.

# FOLLOW-THE-CARROT

- The carrot point (CP) is defined as the point on the path that is one look ahead distance L away from the robot.
- The orientation error ε is defined as the angle between the x' axis (pointing forward) and the line from the robot to the carrot point
- If ε = 0 : the robot points towards CP, i.e. the robot should continue straight ahead
- Otherwise, the steering angle φ or angular velocity ω may be controlled to reduce ε:
  - φ = g1 ε
  - ω = g2 ε
  - where g1 and g2 are constant gain factors

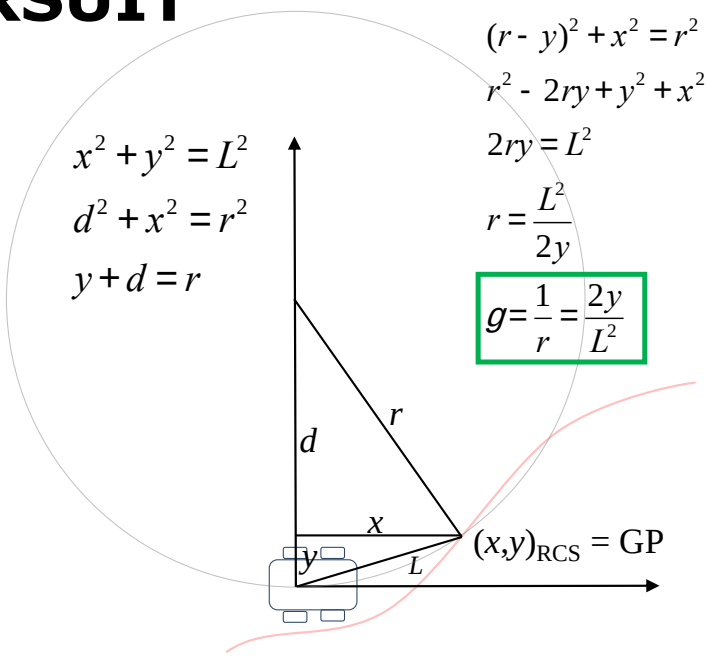


# PURE PURSUIT

- Uses the fact that the robot moves along circles
- Define a goal point (GP) on the path, at distance L from the robot
- Construct a circle passing through $(0,0)_{RCS}$ and GP, such that the vehicle orientation is a tangent to the circle
- The circle is defined by its radius r and midpoint
- Set φ or ω (turing rate) to correspond to motion along this circle:
  - φ (depends on the mechanical design)
  - $\boxed{\omega = v\,\Upsilon,}$
  - where $v$ is the linear speed and $\Upsilon = 1/r$ is the curvature of the circle

$$x^2 + y^2 = L^2$$
$$d^2 + x^2 = r^2$$
$$y + d = r$$

$$d = r - y$$
$$(r - y)^2 + x^2 = r^2$$
$$r^2 - 2ry + y^2 + x^2 = r^2$$
$$2ry = L^2$$
$$r = \frac{L^2}{2y}$$
$$\boxed{g = \frac{1}{r} = \frac{2y}{L^2}}$$

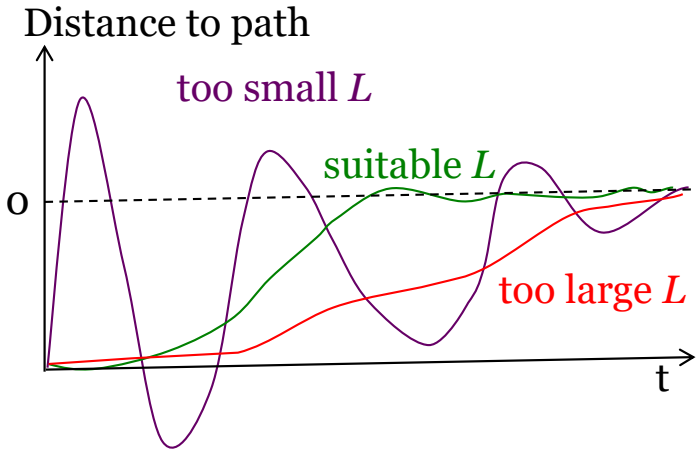


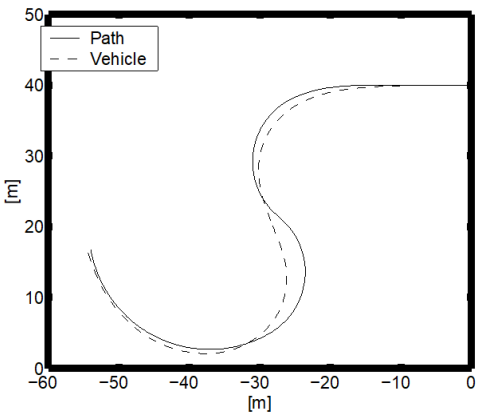

Derivation:
$\omega = 2\pi/T = 2\pi r/Tr = C/Tr = v/r = v\Upsilon$
where $C$ is the circumference and $T$ is the time it takes to move a full turn around the circle

Reference: *Implementation of the Pure Pursuit Path Tracking Algorithm*, R. Craig Coutier, 1992 (note: different orientation of the robot…)

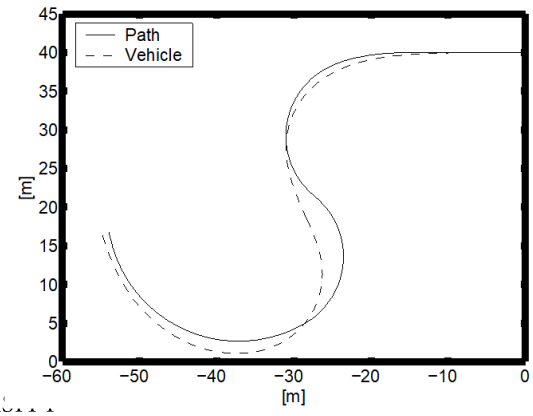
# PERFORMANCE

- Pure pursuit is regarded to be better (e.g. works better when the robot is far off the path)
- Quite similar performance in practice
- Both methods oscillate or are slow to reach the path, depending on $L$
- $L$ may be modified as a function of speed (Urmson et al. 2004)
- General problem: Cutting corners:




Pure pursuit



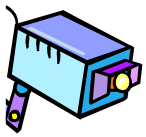

Follow-the-carrot

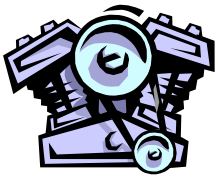
# ROBOT ANATOMY (PART 1)

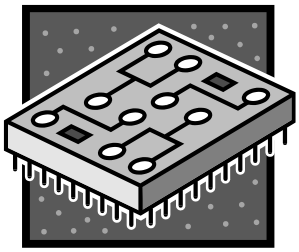
**"Anatomy is the study of the structure of the body and its parts"**

UMEÅ UNIVERSITY

# WHAT CONSTITUTES A ROBOT?





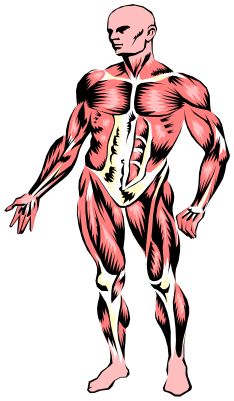


**Hardware:**

- Sensors
- Effectors/Actuators
- Control unit

**Software:**

- System software
- Application software



# SENSORS

are used by the robot to gain information about the environment (and itself) to guide control

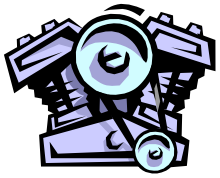
**Some typical usages:**

- Avoid collision (distance sensors, “bumpers”)
- Identifying objects (cameras)
- Localization (GPS, odometry)
- Building maps (distance sensors)
- Feedback control (joint sensors)
- …

# EFFECTORS & ACTUATORS

**Effector**

- A device by which a robot has an impact on the environment. Most common two types: *locomotion* and *manipulation*.
- Human effectors: Arms, hands, legs
- Robot effectors: wheels, tracks, grippers, ...



**Actuator**

- A mechanism that enables the effector to execute an action.
- Human actuators: Muscles
- Robot actuators: Electric motors, hydraulic/pneumatic cylinders, solenoids

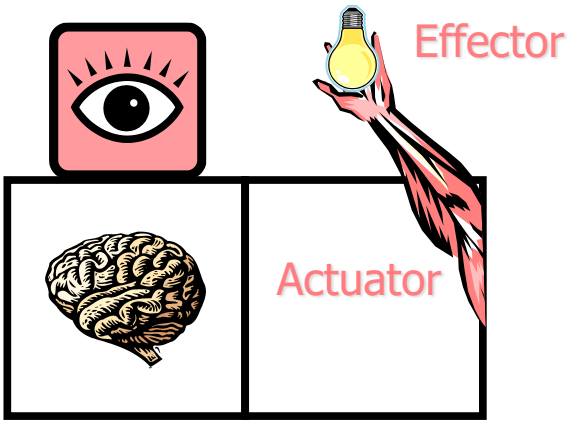
# CONTROL UNIT

- Uses sensor data to compute control signals to the actuators that move the effectors
- The computation depends on the task:
  - Move a robot arm in a fixed pattern (e.g. spray painting)
  - Look for humans (shop assistant robots)
  - Find the shortest way out of the building (rescue robots)
  - Do the laundry!
  - ...
- Human control units: A brain
- Robot control units: Most often a computer

# Putting it all together:
## Two implementations

**A human being**

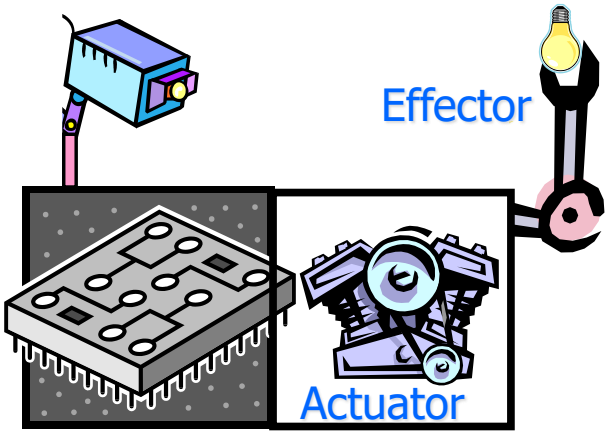
Sensor




Control unit

**A robot**

Sensor




Control unit

**We are trying to mimic nature!**

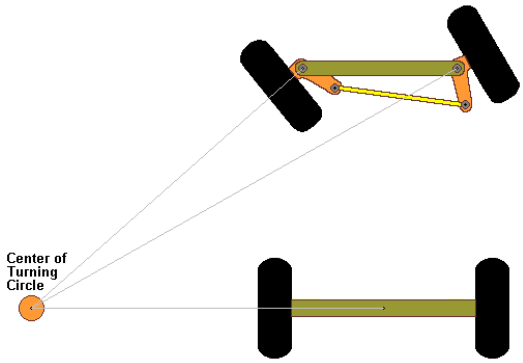
UMEÅ UNIVERSITY

# COMMON EFFECTORS FOR PROPULSION

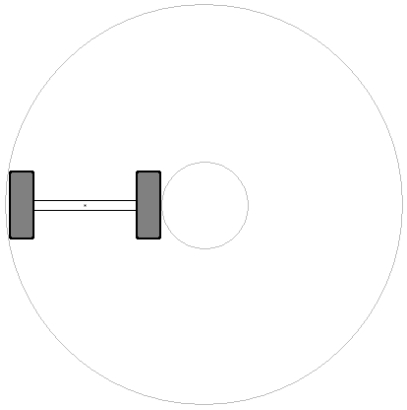
## Wheels!

- Linear motion – easy: set left and right wheel speed equal
- Turning – several approaches:

Car-like

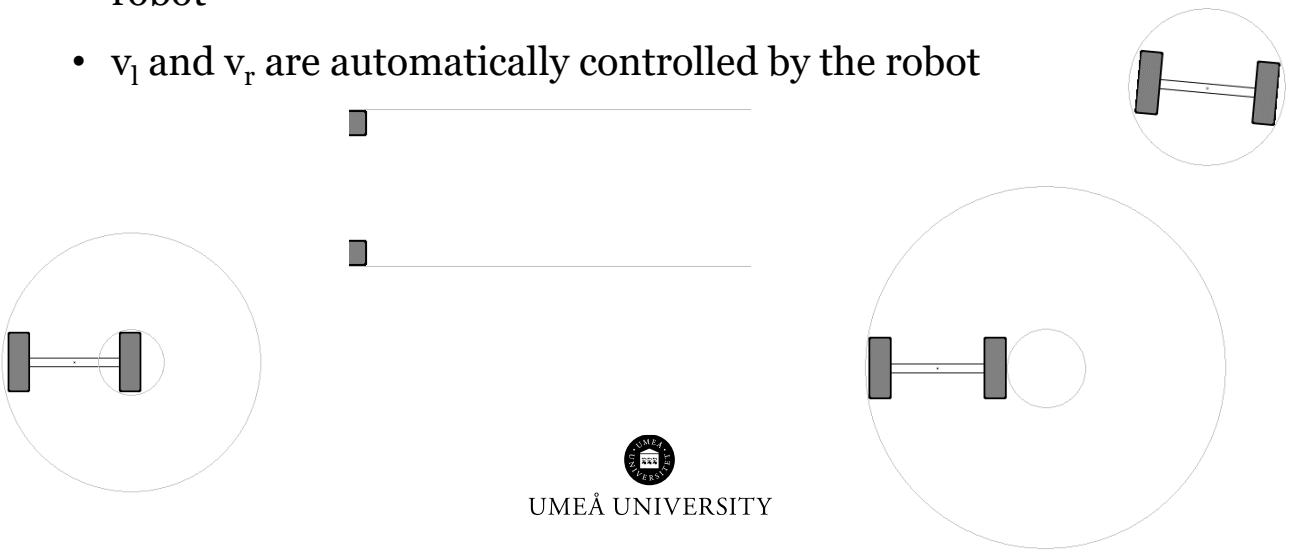



Differential



# DIFFERENTIAL STEERING

- Very common technique for mobile robots
- Turning by changing the relation between $v_l$ and $v_r$ (speed of left and right wheel)
- The Kompai robot is controlled by specifying linear velocity v and angular (rotational) velocity ω.
- ω (radians/seconds) is the rate of change of the heading of the robot
- $v_l$ and $v_r$ are automatically controlled by the robot



# MORE ON SENSORS

- Interaction with the environment is what makes Robotics special
- Sensors are essential!
- We will try to answer the following questions:
  - What is a sensor?
  - What are the most typical sensors?
  - What is sensor fusion?

# WHAT IS A SENSOR?

A sensor measures a particular *physical entity* that has a known relationship with *a property of the environment (or robot)*.

**Example:**
A speedometer in a car measures the number of revolutions of the wheels per unit of time.

- Assumption: the number of revolutions is proportional to the speed of the car.

# WHICH PROPERTY OF THE ENVIRONMENT ?

- **Proximity sensors**: Direction and distance of objects around the robot

pose
- **Position sensors**: (x,y,z) for the robot
- **Attitude sensors**: (roll, pitch, yaw) for the robot

- **Speed**: Change in pose
- Ambient light level
- Temperature
- Identity of objects
- ...

# PROXIMITY SENSORS

# PROXIMITY SENSORS

Delivers direction and distance to objects (obstacles, goals, ...)

**Range sensors:**

- Infrared (IR)
- Ultrasonic sonars (or  simply Ultrasonics)
- Laser Scanner
- 3D Cameras

**Contact sensors:**

- Bump and tactile sensors

# REFLECTIVE IR SENSORS

Physical entity

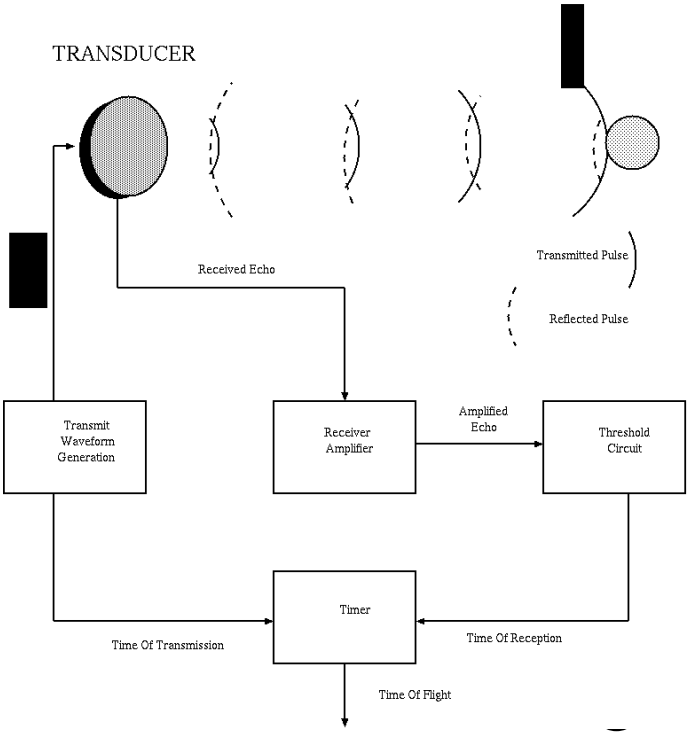
- Emits IR-light and measures the intensity of the reflexion
- Intensity ⇒ Distance

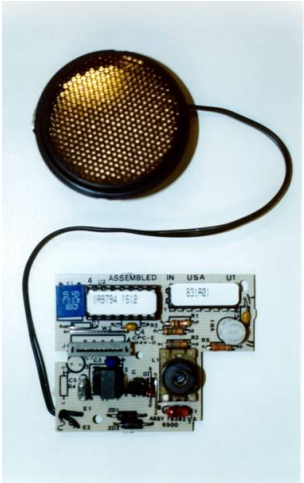
Property of the environement

- Sensitive to ambient light
  - Solution: Modulated IR
- Sensitive to varying absorption of light
  - Depends on the material

# ULTRASONICS (SONAR)

- Emits a 50 KHz “chirp” and “listens” for bounce back
- Determines range based on time of flight



Property of the environement

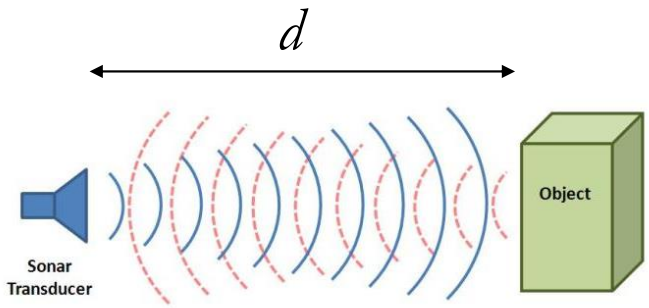
# DETERMINING DISTANCE *d* FROM TIME OF FLIGHT *t*

Physical entity

$d = ½\, c\, t$

$c = c_o + 0.6\, T$

$c_o = 331\ m/s$




Basic sonar illustration – a transducer generates a sound pulse and then listens for the echo.

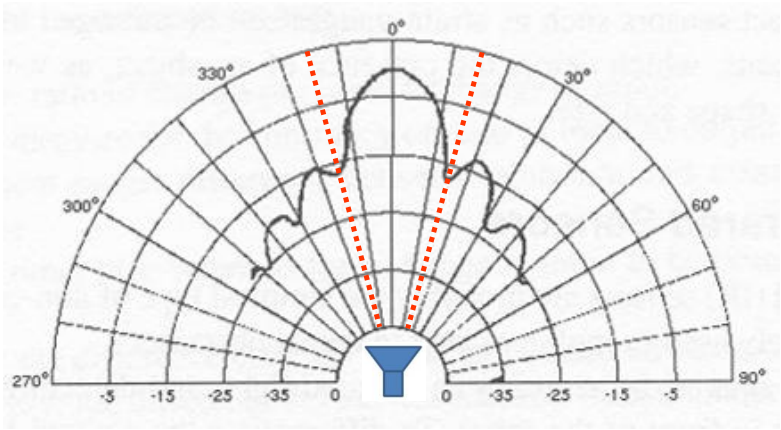
$T$ : temperature in degrees Celsius

$c_o$ : speed of sound at 0° Celsius

**Given *d* : Where is the Object?**

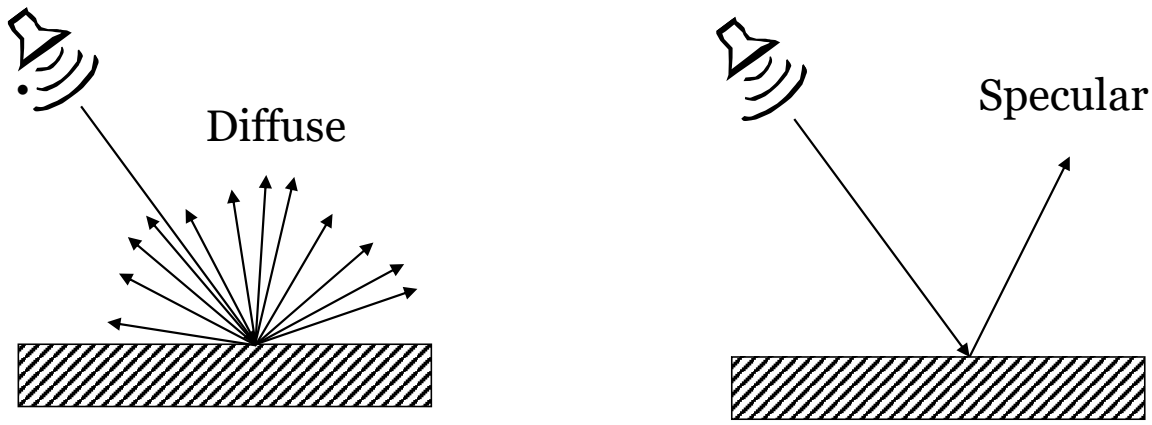
# UNCERTAINTY IN ULTRASONIC SONARS

Sensitivity as a function of angle (logarithmic scale )




- The *main lobe* is pretty well defined as +/- 15°
- Assumption: all echoes come from the main lobe
- The exact location of the echo is still uncertain

# TWO MAJOR KINDS OF REFLECTION

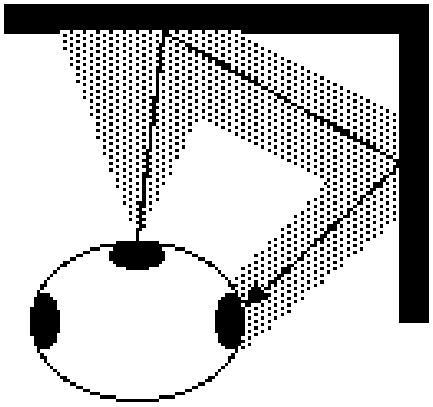



- Diffuse is ok:
  - High probability that an echo will return
- Specular is no good:
  - The sound may return later or not return at all. The former is harder to deal with.

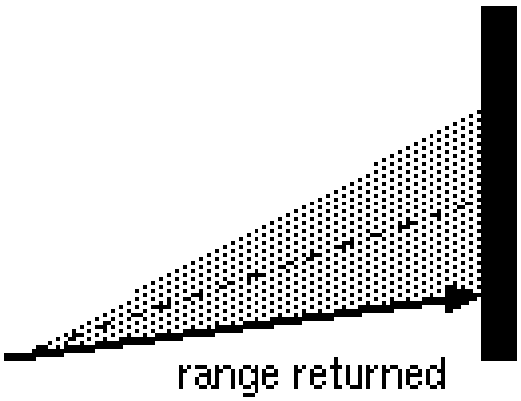
Hard shiny surfaces are problematic for sonars (and rough ones)

# OTHER SOURCES OF UNCERTAINTY

Cross talk



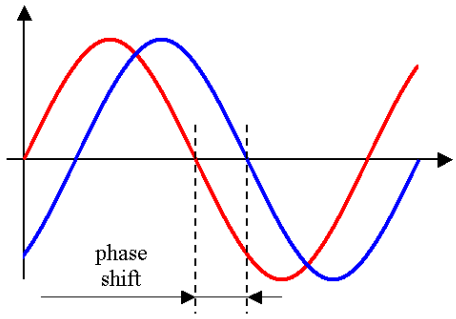
Foreshortening



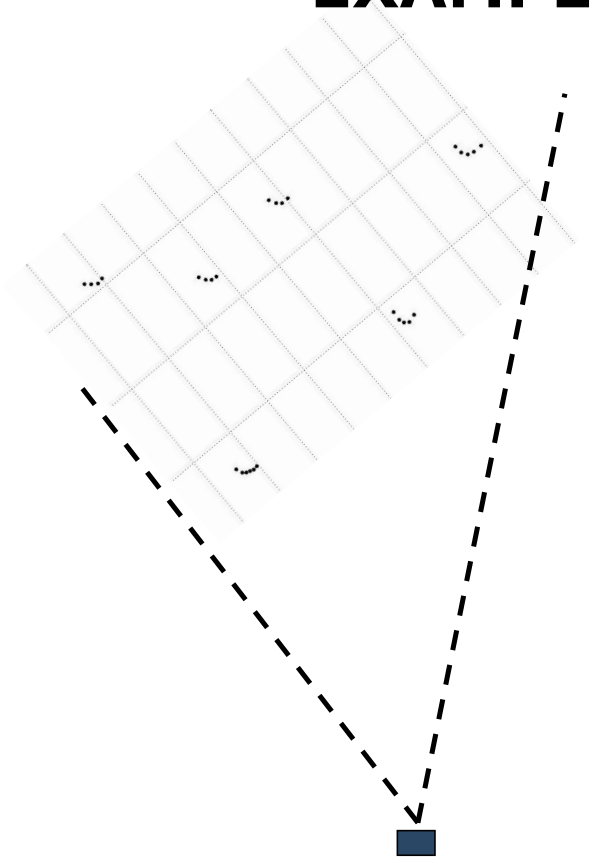
# LASER SCANNER

Hokuyo
URG-04LX

- Ultrasonic sonars measures the time of flight for an emitted sound burst
- Same technique can be used with a laser beam
- But light is much faster
  - Still: Some laser scanners measure time of flight.
  - Easier to modulate the light and measure phase shift.
- A laser beam is much more narrow than a US main lobe
- In a laser scanner, beams are emitted by a rotating laser, and returns distances for each beam
- Usually harmless, invisible light!
- Specular reflection can still be a problem

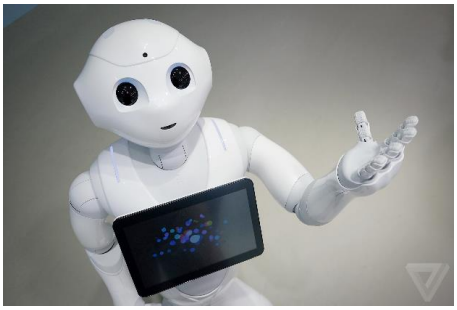


# EXAMPLE OF LASER SCAN

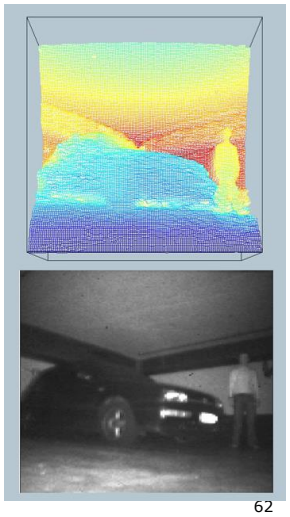



- A scan consists of 361 ranges, one for each 0.5 degree in 0..180.
- From angle and range, (x,y) can be computed

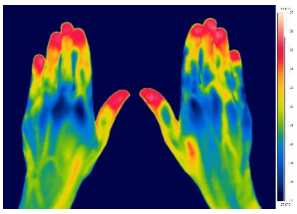
# CAMERAS



- In many respects the most important sensor for humans and animals
- Enables detection, classification and localization of objects and environment
- RGB cameras are most common
- Some cameras extend the capabilities of the human eye
  - 3D cameras



  - Thermal cameras



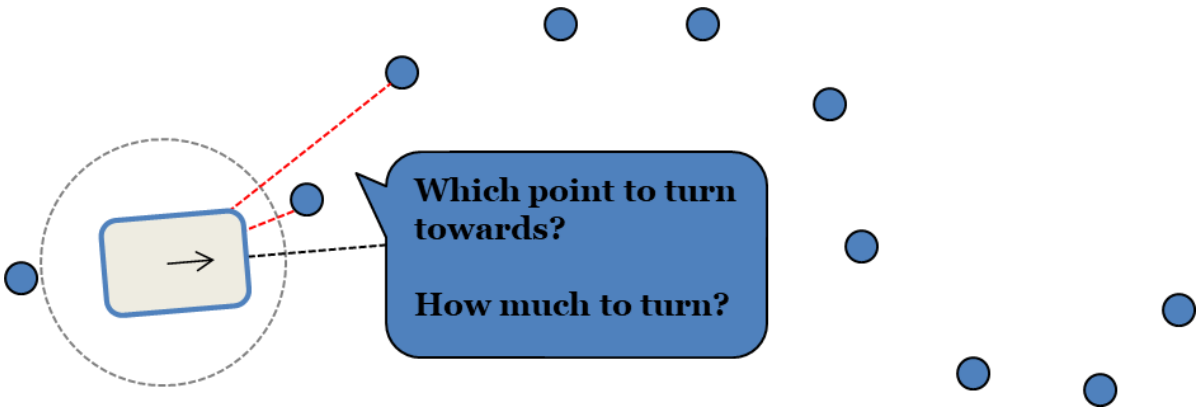
- Require complex image analysis

UMEÅ UNIVERSITY

# ASSIGNMENT 1: FOLLOW THE PATH

# FOLLOW THE PATH

- Create a program that controls the robot in a simulated environment such that the robot follows a specified path
  - Use Follow the Carrot or Pure Pursuit
  - See course planning on Canvas for some Math tips



# FOLLOW THE PATH

- You will work with a simulated robot, in Microsoft Robotic Developer Studio 4 (MRDS)
  - Use the office environment
- Create a program that can read sensors of the robot, and post commands to it
  - Communicate with the robot over HTTP
  - Helper functions contain all low-level communications
  - Example programs: TestRobot3.py, TestRobot3.java
  - Python code also include plotting

# EXAMINATION OF ASSIGNMENT 1

- Work in pairs, **demonstrate** your working solution **on Zoom**
  - You will get a new path for the demo
  - We will create a sign-up list on Canvas
- After the demonstration, **compare** your results to another group and write a **1 page report** about similarities/differences together
- Demo: 2020-09-16
- Report: 2020-09-17

# NEXT LECTURE

- Next lecture: The intelligent agent with Andrea ☺

# THE INTELLIGENT AGENT

## Fundamentals of Artificial Intelligence

UMEÅ UNIVERSITY

# THE INTELLIGENT AGENT

?

## Fundamentals of Artificial Intelligence

UMEÅ UNIVERSITY

# THE INTELLIGENT AGENT

? ?

## Fundamentals of Artificial Intelligence

UMEÅ UNIVERSITY

# LECTURE PLAN

1. What are agents?
2. What are intelligent agents?

3. Types of task environment
   BREAK
4. Types of agents

5. Representing the world

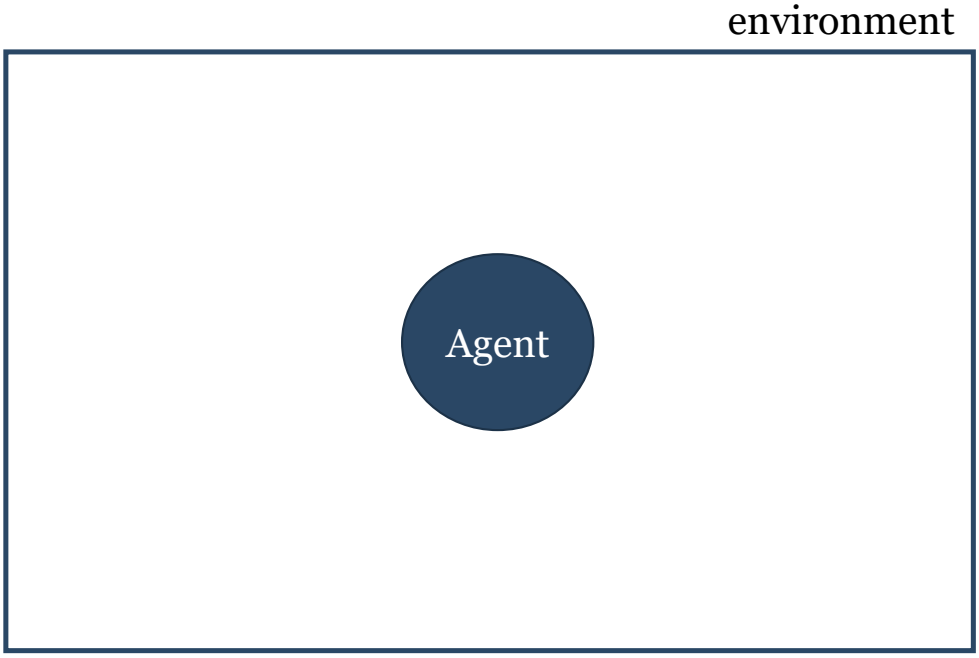
PLEASE GO TO: www.menti.com CODE:13 94 26 9

UMEÅ UNIVERSITY

# 1. WHAT ARE AGENTS?

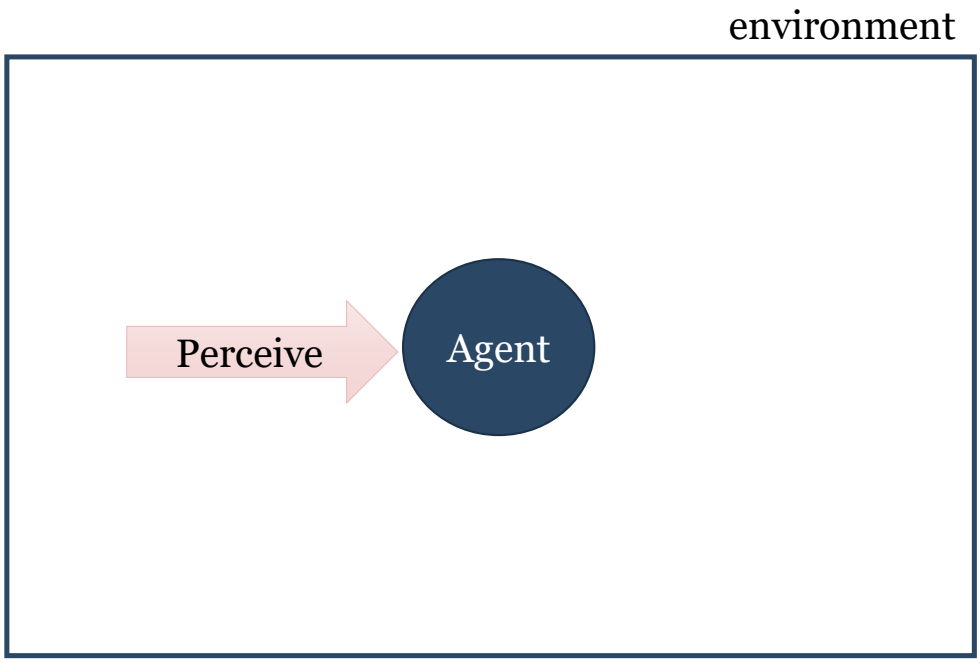
UMEÅ UNIVERSITY

# AGENTS & ENVIRONMENT



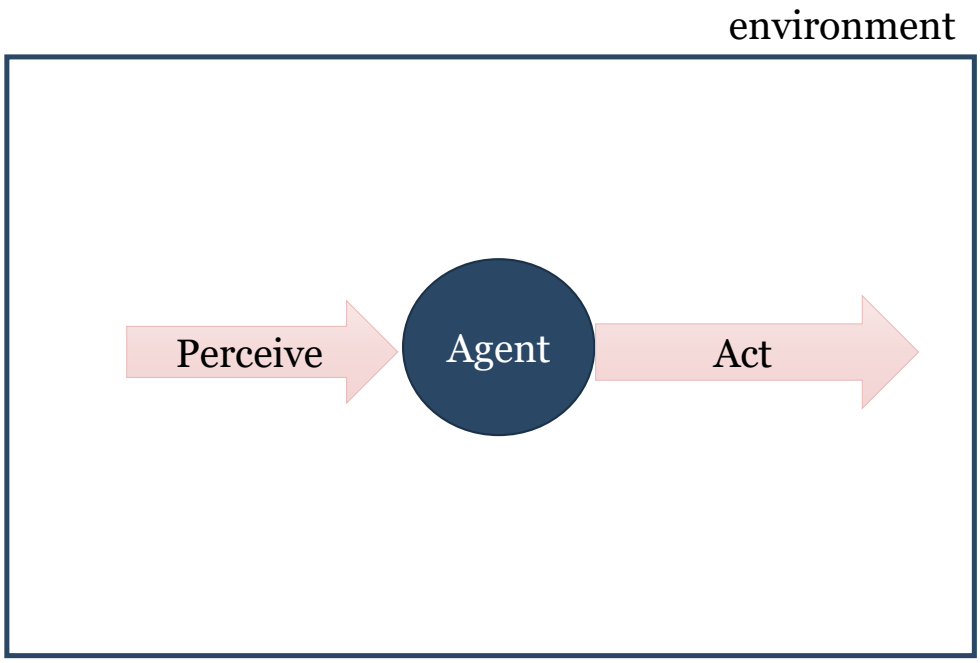

UMEÅ UNIVERSITY

# AGENTS & ENVIRONMENT



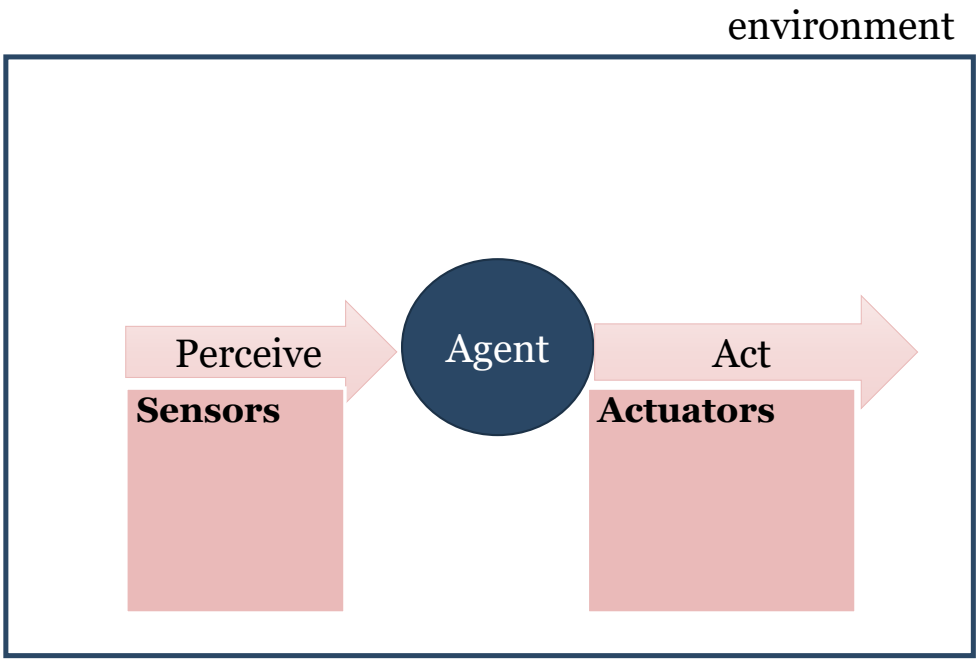

UMEÅ UNIVERSITY

# AGENTS & ENVIRONMENT



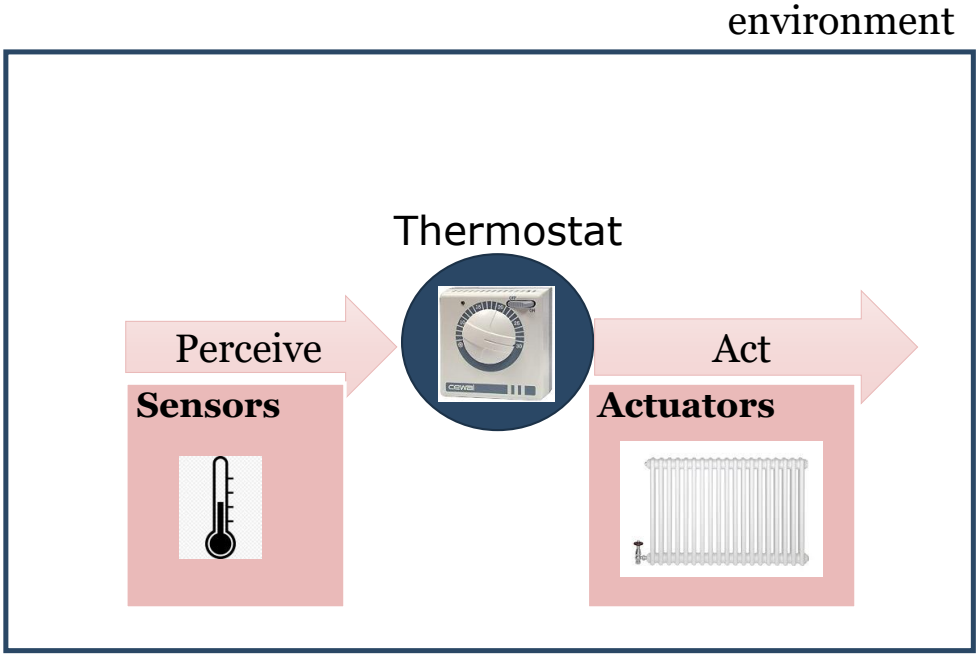

UMEÅ UNIVERSITY

# AGENTS & ENVIRONMENT




UMEÅ UNIVERSITY

# AGENTS & ENVIRONMENT

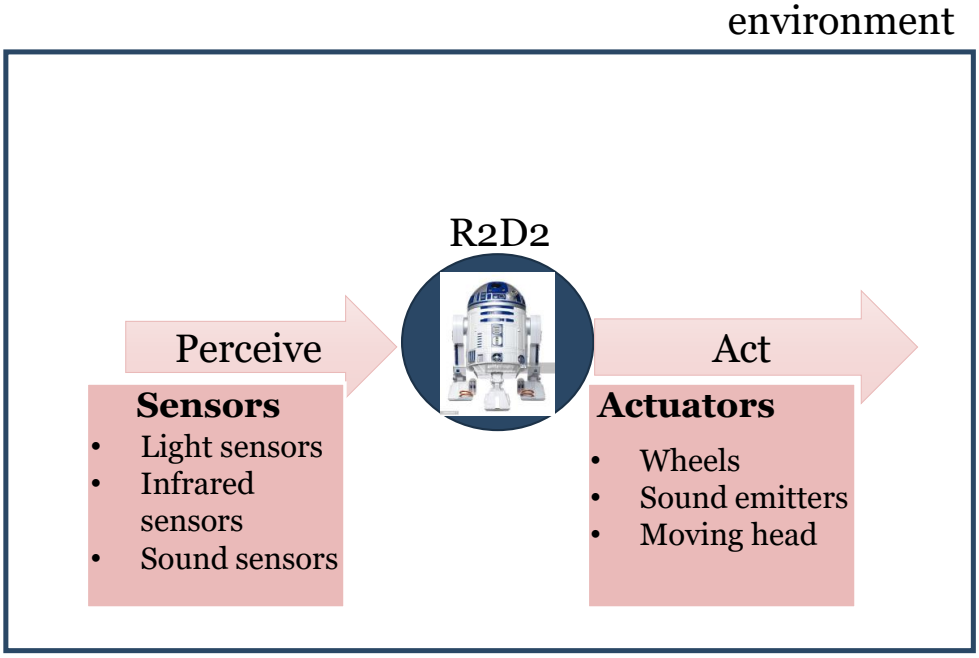



UMEÅ UNIVERSITY

# CAN YOU THINK OF SOME AGENTS?

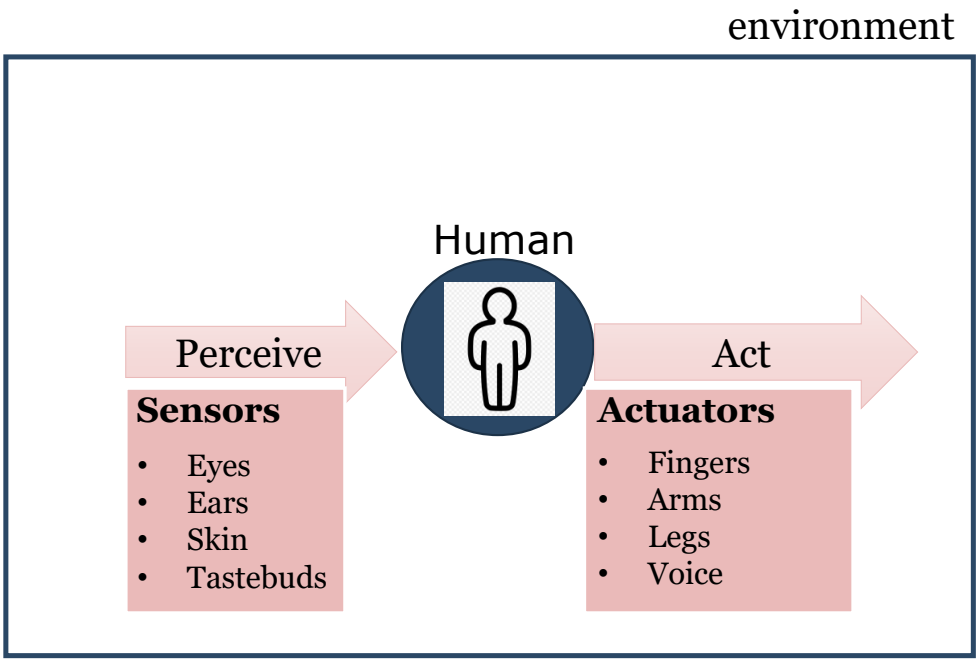
UMEÅ UNIVERSITY

# AGENTS & ENVIRONMENT




UMEÅ UNIVERSITY

# AGENTS & ENVIRONMENT




UMEÅ UNIVERSITY

# HOW CAN WE REPRESENT AGENTS?

- Ideally, an intelligent agent's actions depend on **what it has perceived**.

Anything it has ever perceived=**percept sequence**

What it is perceiving at a specific instant=**percept**

- So, if we know which action the agent will take for **any posible percept sequence**, we know **everything** about the agent.

# THROUGH A FUNCTION

We can describe the agent with an **<u>agent function</u>:**

$$u = f(y)$$

action

percept sequence

This is an **external** representation: it does not tell us *how* the agent computes the actions internally.

# THROUGH THEIR PROGRAM

In an artificial agent, these actions are implemented through an **agent program** that is running.

**Function** *Agent*(location, status) **Returns** action {

process inputs;

update the internal representation;

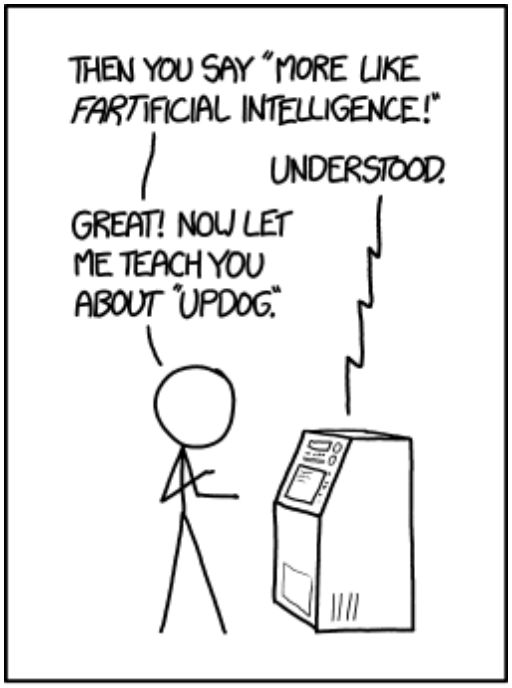
construct a plan;

**return** action;

}

The agent function describes the agent, and the agent program is an implementation of the agent.

# 2. WHAT ARE *INTELLIGENT* AGENTS?




AI TIP: TO DEVELOP A COMPUTER WITH THE INTELLIGENCE OF A SIX-YEAR-OLD CHILD, START WITH ONE AS SMART AS AN ADULT AND LET ME TEACH IT STUFF.

Source: https://xkcd.com/1696/

# RATIONAL AGENTS

*Always doing the right thing.*

?

UMEÅ UNIVERSITY

# RATIONAL AGENTS

*Always doing the right thing.*

?

environment → percepts → actions → changes in the environment

UMEÅ UNIVERSITY

# RATIONAL AGENTS

*Doing the right thing.*

?

environment → percepts → actions → changes in the environment

Good changes?

Bad changes?

UMEÅ UNIVERSITY

# RATIONAL AGENTS

*Doing the right thing.*

?

environment → percepts → actions → changes in the environment

Good changes? Bad changes?

To know what the right action at the right time is, we check if we change **the environment** in the way that we want it to change.

# PERFORMANCE MEASURE& RATIONALITY

A performance measure **evaluates any given sequence of environment states**.

A performance measure is a **choice by the designer**!

A rational agent is an agent that, for each possible percept selects an action that is expected to maximize its performance measure.

# PERFORMANCE MEASURE

**There isn't one fixed performance measure for all tasks, agents and environments.**

Let's imagine we are designing an autonomous vacuum cleaner.

POLL: WHAT COULD BE A GOOD PERFORMANCE MEASURE THAT THE AGENT NEEDS TO MAXIMISE?

# PERFORMANCE MEASURE

**There isn't one fixed performance measure for all tasks, agents and environments.**

Let's imagine we are designing an autonomous vacuum cleaner.

POLL: WHAT COULD BE A GOOD PERFORMANCE MEASURE THAT THE AGENT NEEDS TO MAXIMISE?

As a piece of advice, R&N say: *As a general rule, it is better to design performance measures according to what one actually wants in the environment, rather than according to how one thinks the agent should behave.*

# PERFORMANCE MEASURE

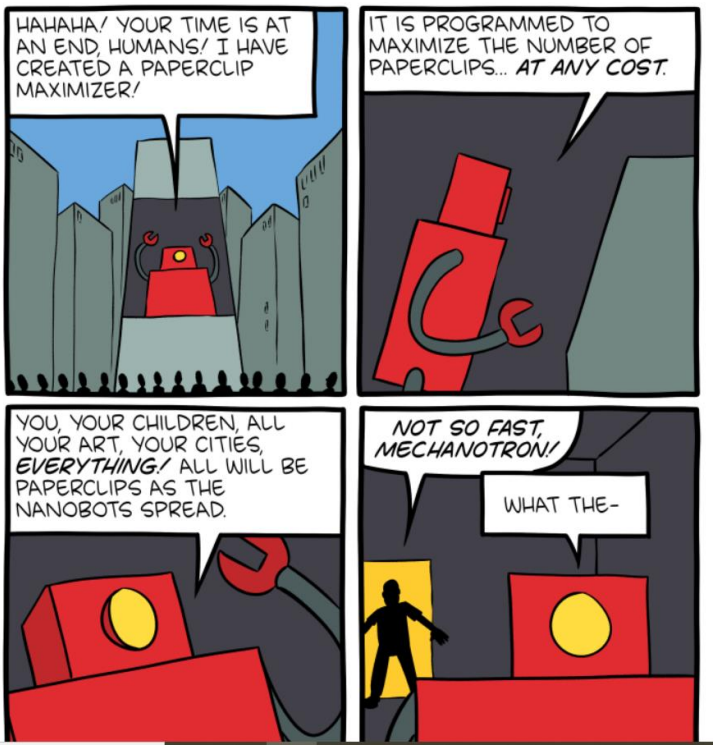
- For the same performance measure, different strategies can be equally effective.

- **Ex: Vacuum at low power all day, versus full power for 5 mins every hour, or at the end of the day.**

- With respect to the amount of clean square meters performance measure, both these strategies are equally good.

- We can of course add elements to our measure to distinguish strategies that we prefer, like electricity consumption.

# PERFORMANCE MEASURE




Source and rest of the story: https://www.smbc-comics.com/comic/paperclip

Based on the paperclip maximiser *thought experiment* by Nick Bostrom:
https://en.wikipedia.org/wiki/Instrumental_convergence

UMEÅ UNIVERSITY

# RATIONAL AGENTS

*Doing the right thing.*

A **rational agent** is an agent that, for each possible percept selects an action that is expected to maximize its performance measure, given the evidence provided by the percept sequence and whatever built-in knowledge the agent has.

# RATIONAL AGENTS

*Doing the right thing.*

A rational agent is an agent that, for each possible percept, selects an **action** that is expected to maximize its **performance measure**, given the evidence provided by the **percept sequence** and whatever **built-in knowledge** the agent has.

# RATIONAL AGENTS

## *Doing the right thing.*

A rational agent is an agent that, for each possible percept, selects an **action** that is expected to maximize its **performance measure**, given the evidence provided by the **percept sequence** and whatever **built-in knowledge** the agent has.

The **rationality** of an agent depends on:

1. Possible actions
2. Performance measure
3. Percept sequence
4. Built-in knowledge

# RATIONAL AGENTS

IS THE AGENT GIVEN BY THE PROGRAM BELOW RATIONAL?

The agent doesn't know in which square it will be placed or where the dirt is.

| A | B |
|---|---|
| | |

| Percept | Action |
|---|---|
| A, Clean | Move Right |
| A, Dirty | Clean |
| B, Clean | Move Left |
| B, Dirty | Clean |

**Performance measure:** Number of clean squares
**Knowledge:** Clean squares stay clean.

# RATIONAL AGENTS

IS THE AGENT GIVEN BY THE PROGRAM BELOW RATIONAL?

The agent doesn't know which square it will be placed in and where the dirt is.

| A | B |
|---|---|
| | |

| Percept | Action |
|---|---|
| A, Clean | Move Right |
| A, Dirty | Clean |
| B, Clean | Move Left |
| B, Dirty | Clean |

**Performance measure:** Number of clean squares

**Knowledge: A** is never dirty.

# RATIONALITY IS NOT OMNISCIENCE

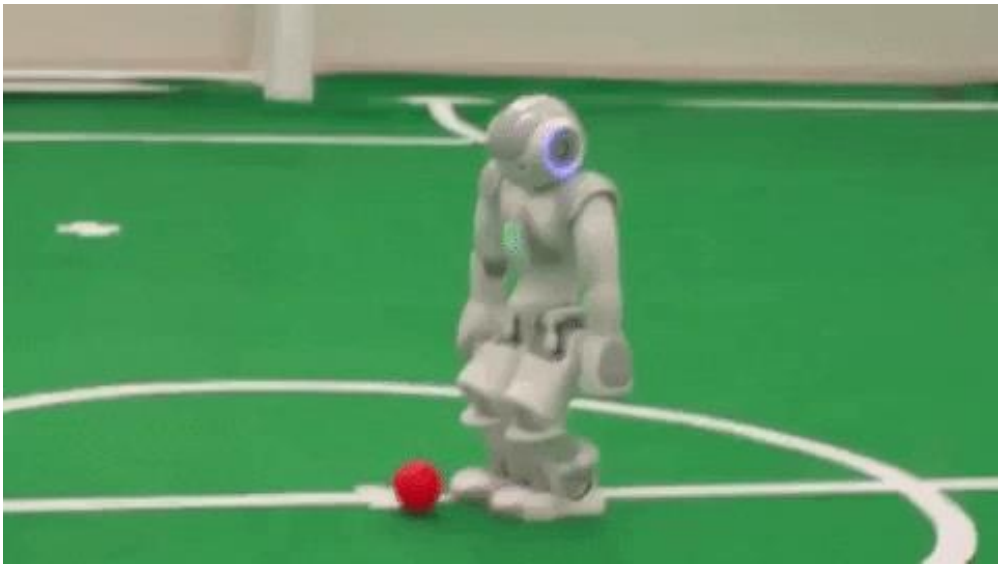
- Rationality depends on the **agent's knowledge.**

- **In most environments, it is imposible to know everything!**

- Rationality is about maximising **expected** performance.

**Gathering information** and **learning** are part of rationality: they change the agents knowledge and can maximise expected performance by adjusting the expectations to reality.

# 3. TYPES OF TASK ENVIRONMENTS



UMEÅ UNIVERSITY

# AGENTS AND THEIR ENVIRONMENT

- The environment in which an agent **acts** affects how we should design it.
- There are a number of concepts that we have defined today that are related to the environment:
  - An agent perceives the environment through **sensors**, and acts on it through **actuators**.
  - The changes in the environment are evaluated with a **performance measure**.

- We call this collection the **task environment**, and it can be described by

**PEAS** (**P**erformance, **E**nvironment, **A**ctuators, **S**ensors)

# TYPES OF TASK ENVIRONMENT

Task environments are hugely varied (since they depend on the deployment context and the agent itself).

Still, we can look at some **properties**:

1. Partly vs. fully observable
2. Static vs. dynamic
3. Discrete vs. continuous
4. Episodic vs. sequential
5. Deterministic vs. stochastic
6. Single vs. multi agent
7. Known vs. unknown

# FULLY VS PARTIALLY OBSERVABLE

- A task environment is fully observable if the agents' sensors detect all aspects that are **relevant** to the choice of action.
  - Depends on the **sensors,** the **environment** and the **performance measure** (what is *relevant*?).
  - *Eg. A thermostat.*

- Otherwise, it is **partially observable**.
  - *Eg. An autonomous vehicle (cannot see objects behind a corner).*

- If there are no sensors, the task environment is **unobservable**.

# KNOWN VS UNKNOWN

- This is a property of **the agent**, rather than of the environment.

- In a **known** environment, the *laws* of the environment are part of the knowledge of the agent.
  - The outcomes or outcome probabilities (*how an action will modify the environment*) are known.
  - This doesn't mean the environment is fully observable (*card game*).

- In an **unknown environment,** the agent does not know the outcome of its actions.
  - *Eg. A game for which the agent does not know the rules.*

# STATIC VS DYNAMIC

- If the environment can change while the agent is deliberating (deciding on an action), it is **dynamic.**
  - *Eg. A cooking robot, where the food can burn while deciding whether to switch off the heat or not.*

Source: https://www.nytimes.com/2010/02/24/dining/24robots.html

- If the world does not change, but the performance score does, the environment is **semidynamic.**
  - *Eg. When the performance measure depends on the time it takes to complete an action.*

- If the agent can take a percept and produce a response without considering the time it takes to produce that response, the environment is **static.**
  - *Eg. A sudoku-solving agent.*

# DISCRETE VS CONTINUOUS

- **Discrete** environments can be divided into a finite number of distinct states.
  - *E.g. A sudoku grid has a finite number of possible states; the vacuum cleaning example from before where the states could be summarized in a table.*
- **Continuous** environments cannot be divided into separate states.
  - The environment may change continuously (*weather conditions*), the agent may be receiving percepts continuously (*infrared sensor*) and/or acting in a continuous manner (*driving*).

- Many environments that are actually continuous are treated as discrete, and viceversa.
  - *E.g. The physical chess board, where time passes continuously.*
  - *Eg. Input from digital cameras is discrete but can be modeled as a continuous smooth input.*

# EPISODIC VS SEQUENTIAL

- An **episodic** task environment is composed of **atomic** (individual) **episodes**.
  - In each **episode**, the agent receives a percept and performs an action in response.
  - What happens in an episode does not affect the next ones.
  - Each episode could be discrete or continuous.
  - *Eg. An agent looking for defective parts in manufacturing.*

- In a **sequential** environment, a decision can affect future decisions.
  - The agent may still ignore previous states in its decision (reactive agent).
  - Can be both continuous (*driving*) and discrete time (*chess*).

# DETERMINISTIC VS STOCHASTIC

- An environment where the next state can be completely determined based on the previous state and the agent's action, is **deterministic**.
  - *Eg. A sudoku-solving agent, the vacuum robot from previous example (because clean squares remained clean).*

- If an environment is not deterministic, it's *stochastic*.
  - Stochastic environments can't be perfectly predicted, but can still be modeled using models that include uncertainty
  - Note that stochastic is not the same as random, a stochastic process depends both on the previous state and a random element
  - *Eg. An autonomous vehicle (pedestrians modify the environment unpredictably).*

# SINGLE VS MULTI AGENT

- In a **single agent** environment, there is a single agent. In a **multi-agent** environment, there are several agents. But… what counts as another agent in the environment?

- If the other entity has a performance measure that **depends on the actions of the first agent**, we are in a multi-agent environment.
  - *Eg. A card game, a fleet of cleaning robots.*
  - **cooperative** (actions can increase all agents' performance), **competitive** (all performances cannot be increased at the same time), or a combination of both.

- Designing multi-agent systems is a whole field by itself.
  - Communication between agents, organizational structures, decision-making processes…
  - Designing a single agent that will operate in a multi-agent environment is also a different challenge: will it need to communicate? Predict other agents' actions?

# IN SUMMARY

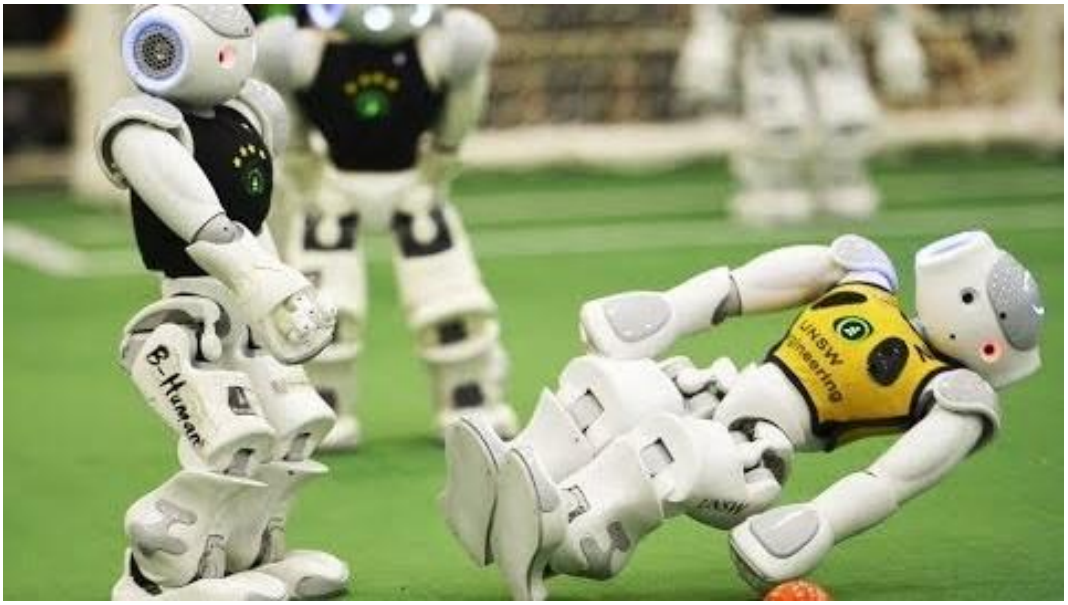
- The task environment is not a description of the environment for a robot, it is **a way to describe the task, or problem, to be solved.**

POLL: What are the characteristics of the following task environment?

A football playing robot.

- Performance measure: Winning with its team.
- Environment: A football field, playing with other robots.
- Acting: Can move and shoot the ball.
- Sensing: Can see the ball, the lines on the field, the other robots, and the goalposts.

# IN SUMMARY




# BREAK

UMEÅ UNIVERSITY

# 4.TYPES OF AGENTS

UMEÅ UNIVERSITY

# AGENT TYPES

1. Simple vs Model-based
2. Reactive
3. Goal based
4. Utility based
5. Learning agents

This classification showcases different ways to design and implement intelligent agents.

# AGENT TYPES

- At the begining of the lecture, we described a vacuum agent with a **table** that contains every posible state of the world, and the corresponding action from the agent.

- This is not possible to do in most cases! The table would be huge.
  - No way to store it.
  - A long time to look it up.
  - A long time to complete it by the designer.
  - A complete knowledge of the state space is needed.

- Effective agent programs also account for **complexity and efficiency**, i.e. they are meant to produce rational agents from a small program.

# SIMPLE & MODEL-BASED AGENTS

- **Simple** agents do not store an internal representation of the state of the world. They react exclusively to the percepts.
  - Great for fully observable scenarios.
  - Eg. thermostat

- **Model-based** agents keep an internal representation of the state of the world, and use it together with the percept to determine the correct action.

- To determine ("best guess") the state of the world, they are endowed with a **model**: information of how the environment evolves around the agent+information of how the agents' actions modify the world.
  - Great for known, partially observable environments.

# REACTIVE AGENTS

- **Reactive** agents select actions on the basis of the *current* percept, ignoring the rest of the percept history.
  - *Eg. The vacuum robot from earlier. Just needs to know if its current square is clean or dirty in order to choose its action.*
  - They do not need memory.
  - Can be described by if-then rules, but in practice there are more sophisticated implementations.

- Pros: Simple, transparent, effective in dynamic, known environments.

# GOAL-BASED AGENTS

- **Goal-based** agents choose their action in order to **achieve a specific goal.**

- Pros: Adaptation to the environment, no need to anticipate all the scenarios, good for dynamic environments.

# GOAL-BASED AGENTS

- When a goal needs a sequence of actions to be achieved in a dynamic environment, it is necessary to deal with hypothetical scenarios: *what will happen if I do this-then-that?*
  - Often, this means that the agent **models** both the environment and the goal.
  - There are also other techniques where the environment is not modeled, only how to act in order to achieve a specific goal.

- Choosing a goal that will maximise the performance measure is also sometimes part of the problem.

- The research areas related to choosing the right actions to execute a goal are those of **search** and **planning**.

# UTILITY-BASED AGENTS

- Sometimes, we cannot capture with a single goal all of the behaviours we want from the agent.
  - For an autonomous vehicle we may want to balance safety, speed to reach destination, electricity consumption, passenger satisfaction…

- **Utility-based** agents choose the action or sequence of actions that are expected to maximise a **utility function.**

- **Utility functions** assign a numerical value to each state of the environment, so that a state the satisfies more preferences has a higher value.

# UTILITY-BASED AGENTS

- Building a utility function can be hard!
  - Can account for different weights between all the factors (more important to not hit pedestrians than to drive fast), but we have to understand what is the right balance.
  - Describing preferences in a way that a variable/number can be assigned is difficult.
  - The agent needs to have a good model of the environment to predict the expected utility of its actions.

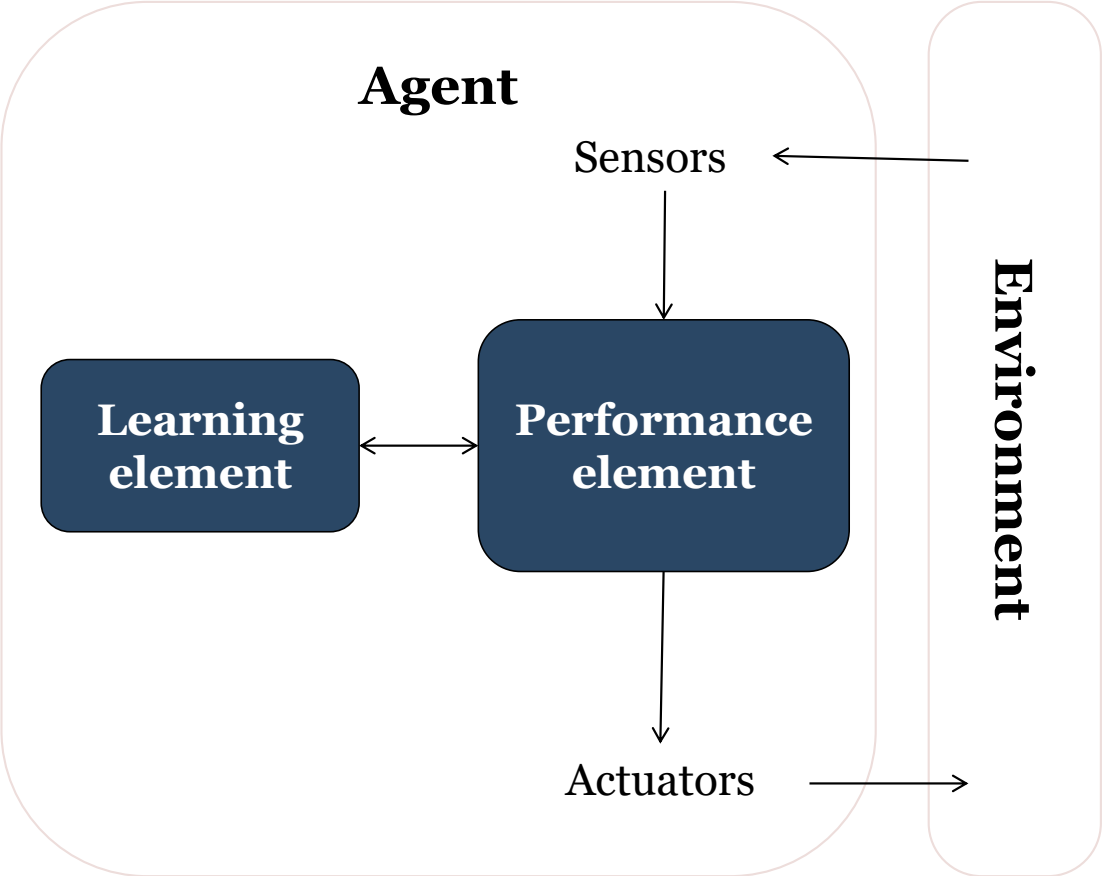
- **Pros:** can capture complex drives, good for unknown environments.

# PERFORMANCE MEASURE, UTILITY AND GOALS

- Notice that both the performance measure and the utility function describe what are good **changes in the environment**.
  - The performance measure is an **external evaluation** of the agent, whereas the utility function is an **internal representation** in the agent.
  - If the utility function is the same as the performance measure, then a utility-based agent will be rational by definition.

- For agents to be rational, the choice of actions, goals or utility function need to be **aligned with the performance measure**.

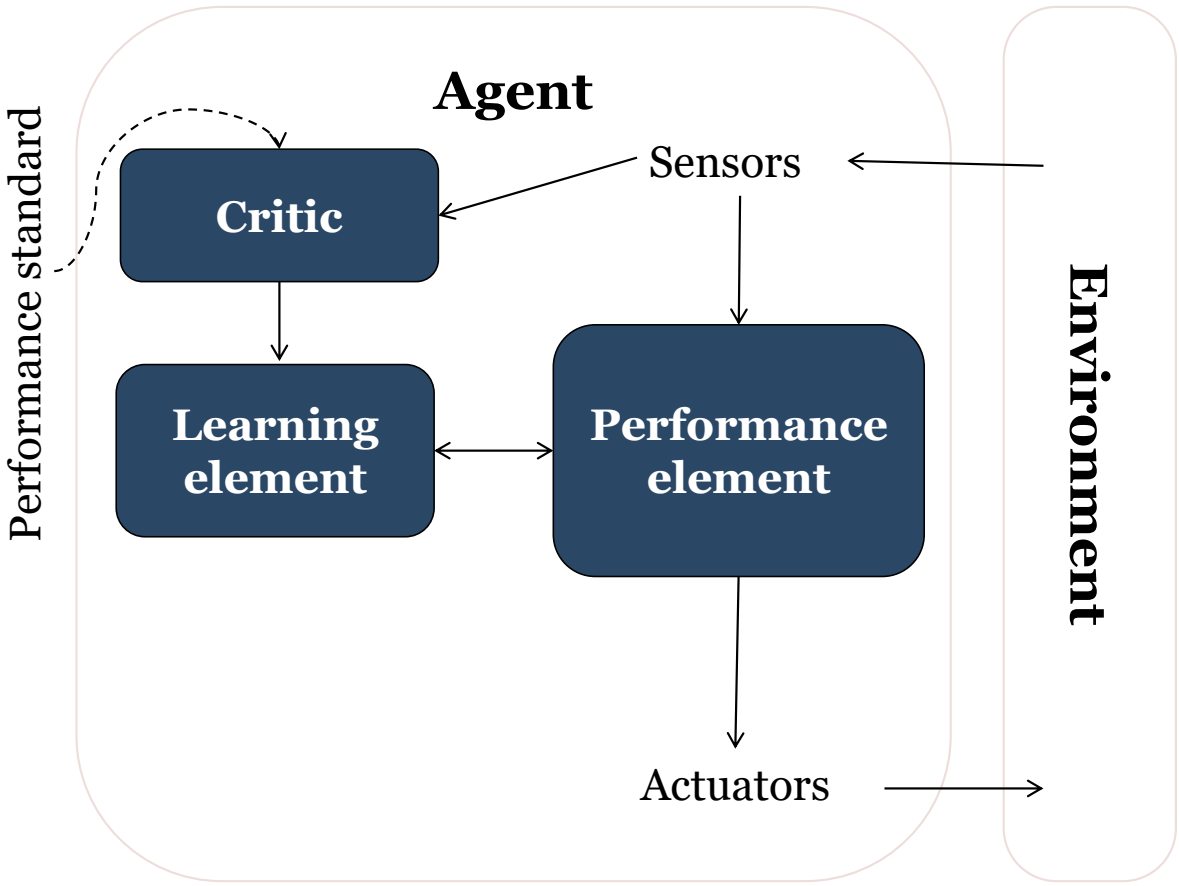
# LEARNING AGENTS






R&N p. 55

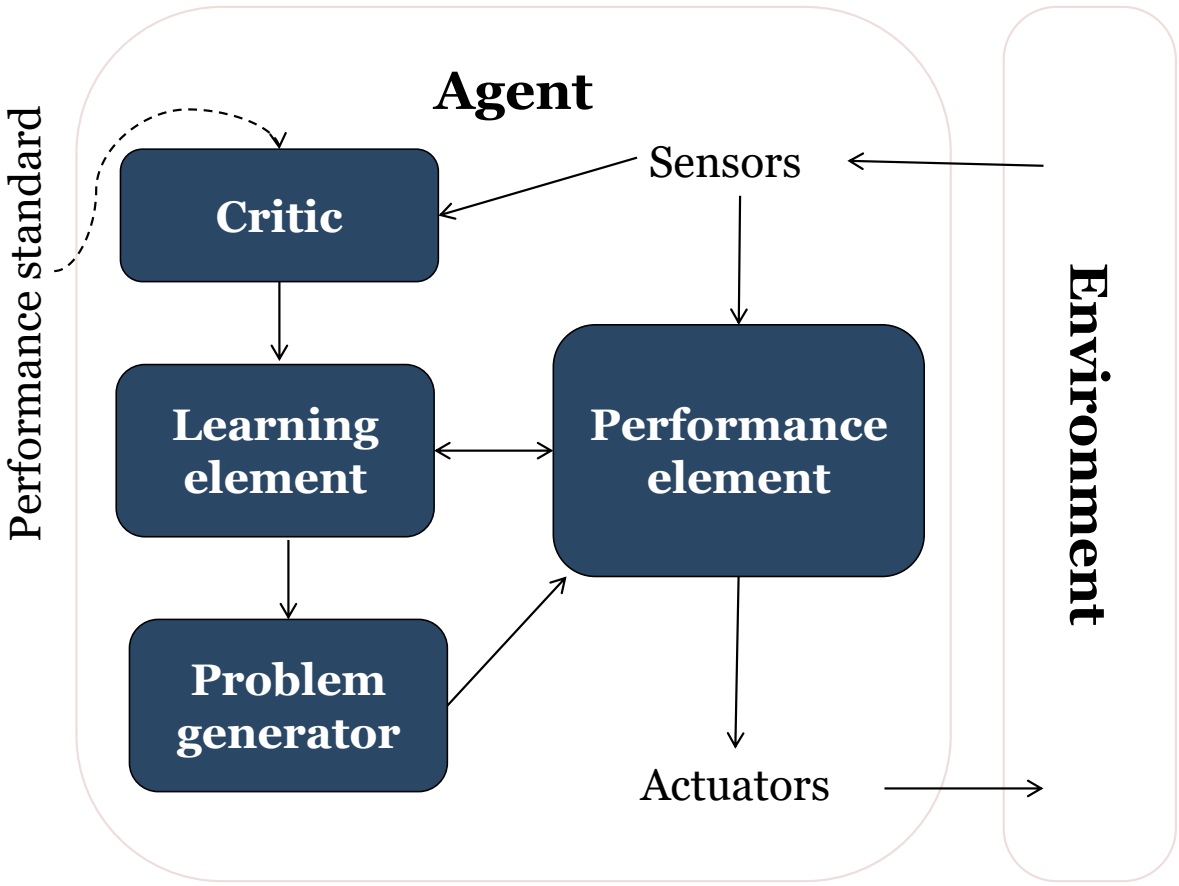
# LEARNING AGENTS






R&N p. 55

# LEARNING AGENTS






R&N p. 55

# WHAT TYPE OF AGENT DO YOU THINK OF WHEN YOU THINK ABOUT AI?

UMEÅ UNIVERSITY

# LEARNING AGENTS

- There are many types of learning agents, but not all of AI is about learning agents.
  - The critic component has to be aligned with the performance measure, which can be hard to implement.
  - Pros: Great for unknown environments, and for task environments with very big state space.

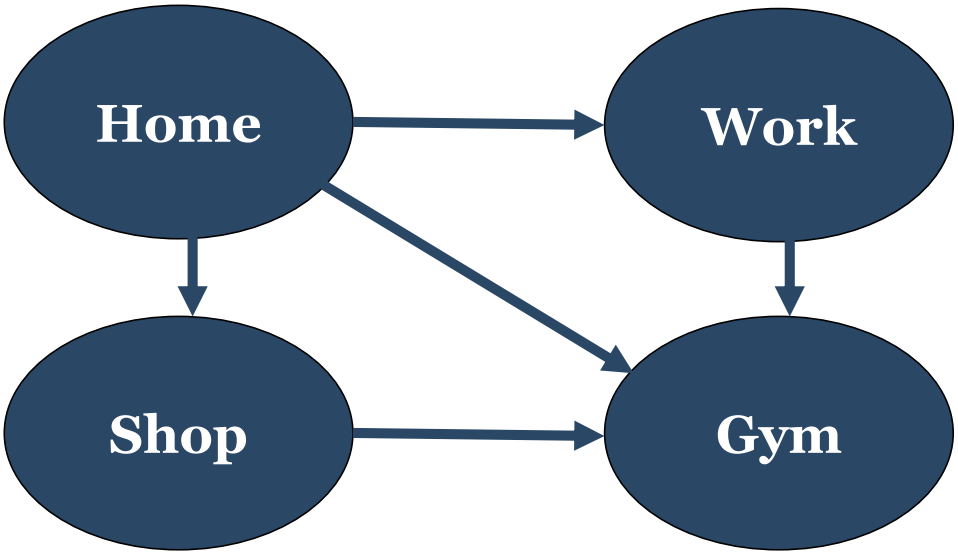
- Agents purely based on reasoning or planning with a full model of the world are also part of AI.

# 5. REPRESENTING THE WORLD

UMEÅ UNIVERSITY

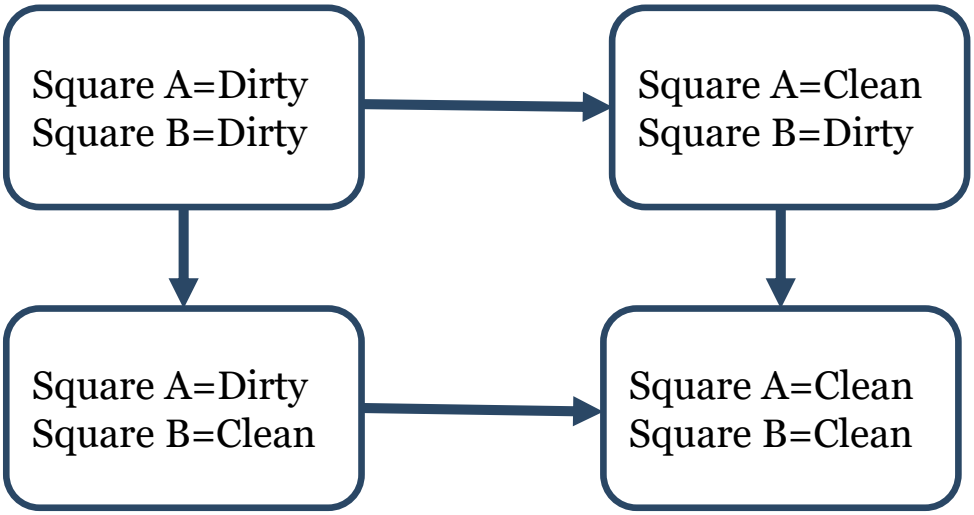
Credit: Disney

# ATOMIC REPRESENTATIONS




- Each state is **a discrete single entity**, we know when it is posible to transition between two states.
- There are **no sub-entities** describing each state, it is just a point in a graph.

# FACTORED REPRESENTATION




- States are described by a fixed set of **variables** or **attributes**, each of which can have a **value**.
- Easier to see how to transition from one state to another (i.e. which attributes need to be changed in order to reach the new state).
- We can represent uncertainty by leaving an attribute blank.

# STRUCTURED REPRESENTATION

State where my dog is waiting behind the door for me to arrive, and he has eaten the cooked chicken that I prepared for dinner.

→

State where I am back home and hungry, but my dog is happy.

- Each state can be described in a model, where we can represent **objects and relationships between them** (e.g like with natural language, or with first order logic).
- This representation is more detailed than assigning values to atributes. We do not need to describe every state with the same atributes, only with the same language.

# REPRESENTATIONS

- We say that a representation in which we can capture more details is more **expressive.**

- A more expressive representation can capture everything a less expressive representation can, and add more details.

# Thank you!

UMEÅ UNIVERSITY

## A few rules while setting up

- **No chatting on zoom**
  - Questions will be answered at specific times
- **Suppress distractions**
  - Turn off your phone and put it out of sight
  - Clear all sources of notifications on the computer
    - Mails, Facebook
  - Open a clean browser with only the google docs with your answers

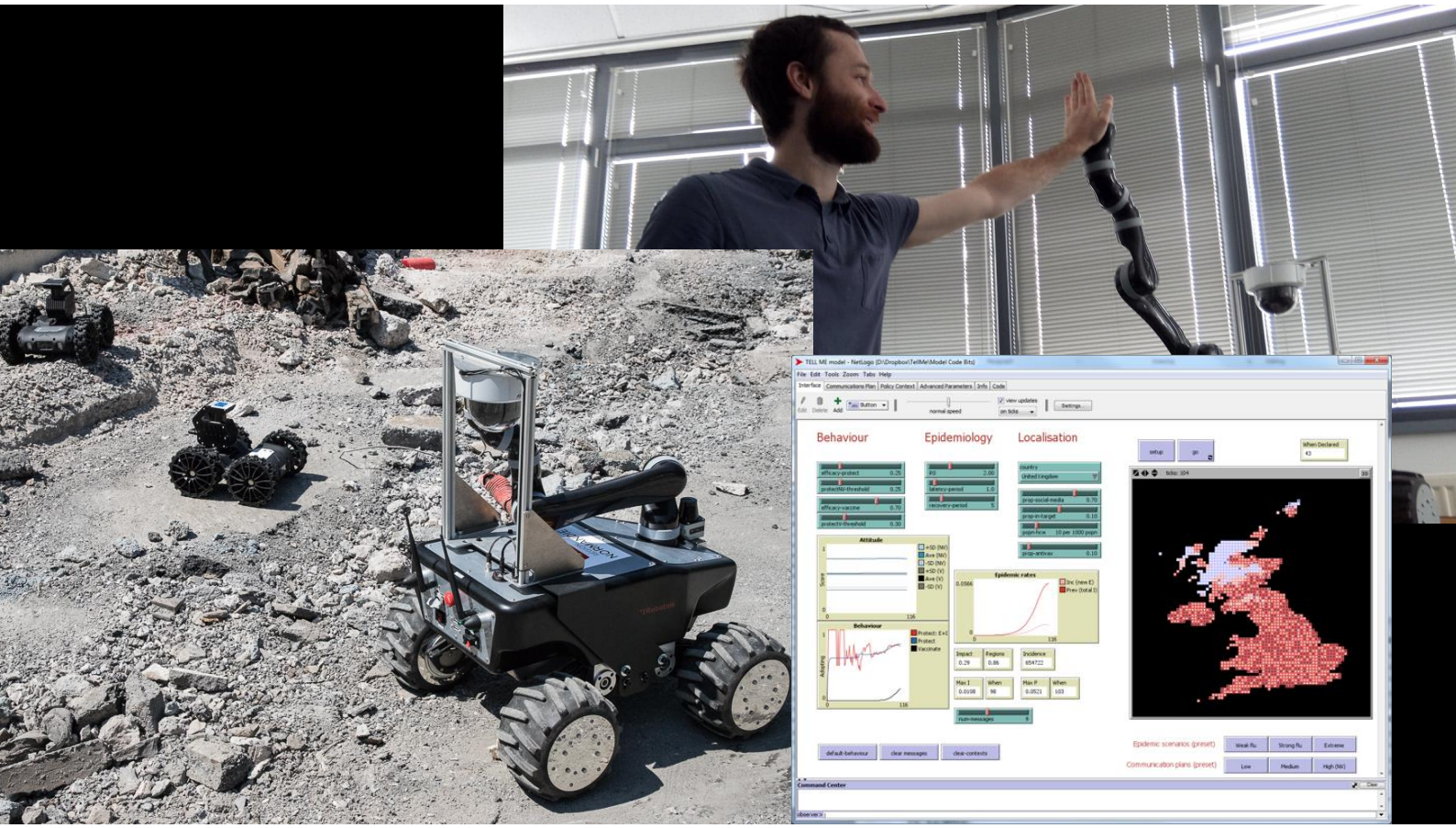
# FUNDAMENTAL PROBLEMS WITH CLASSICAL AI

Loïs Vanhée
Associate professor
Responsible and Ethical Artificial Intelligence
loisv@cs.umu.se

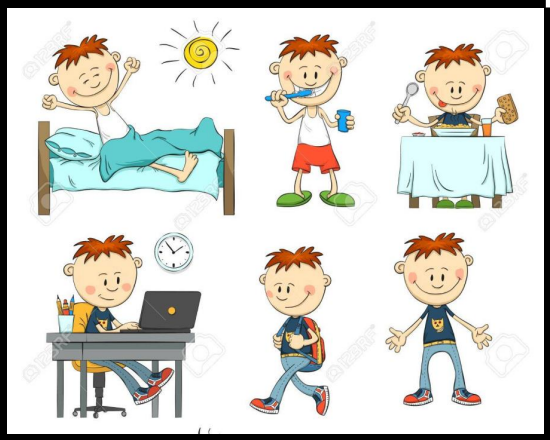
5DV124,5DV201
Fundamentals of Artificial Intelligence

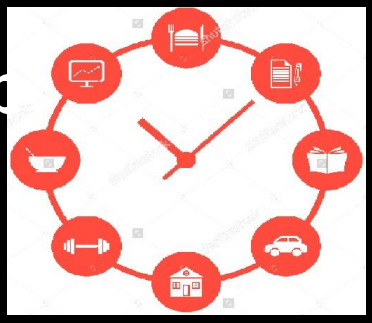
Department of Computing Science

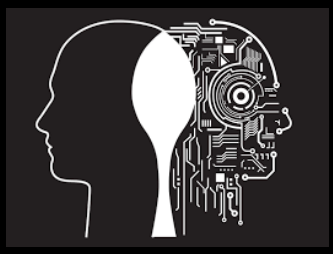
UMEÅ UNIVERSITY


Behaviour
Epidemiology
Localisation
Command Center

# Time to wake up



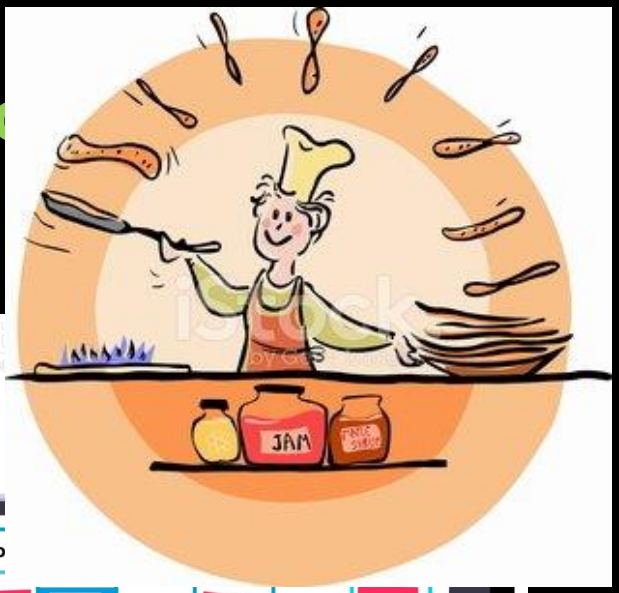


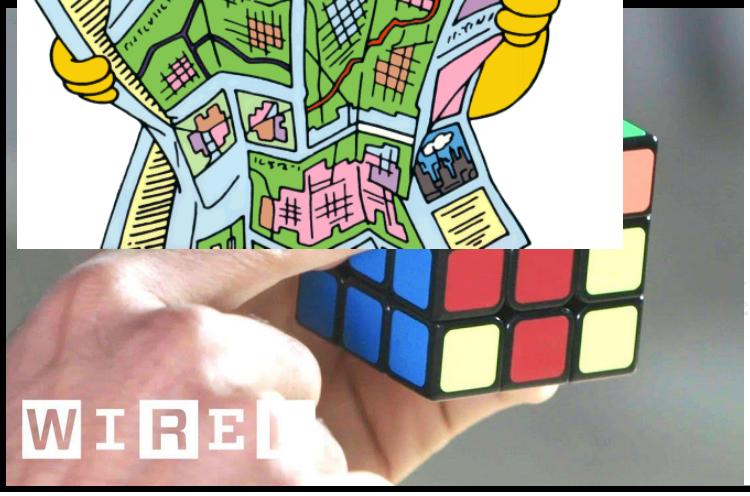


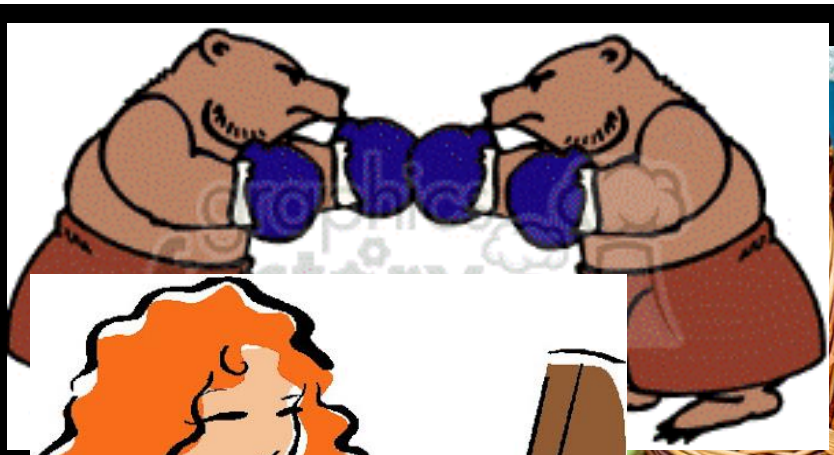
# What is intelligence?

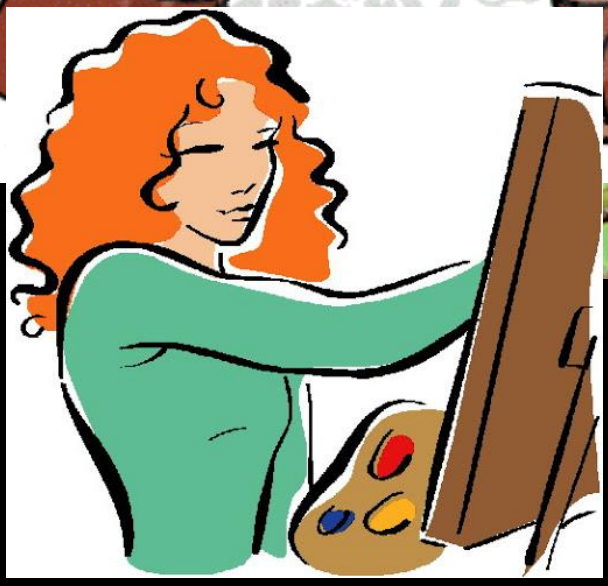
How can intelligence be built or integrated in systems?

olving pr

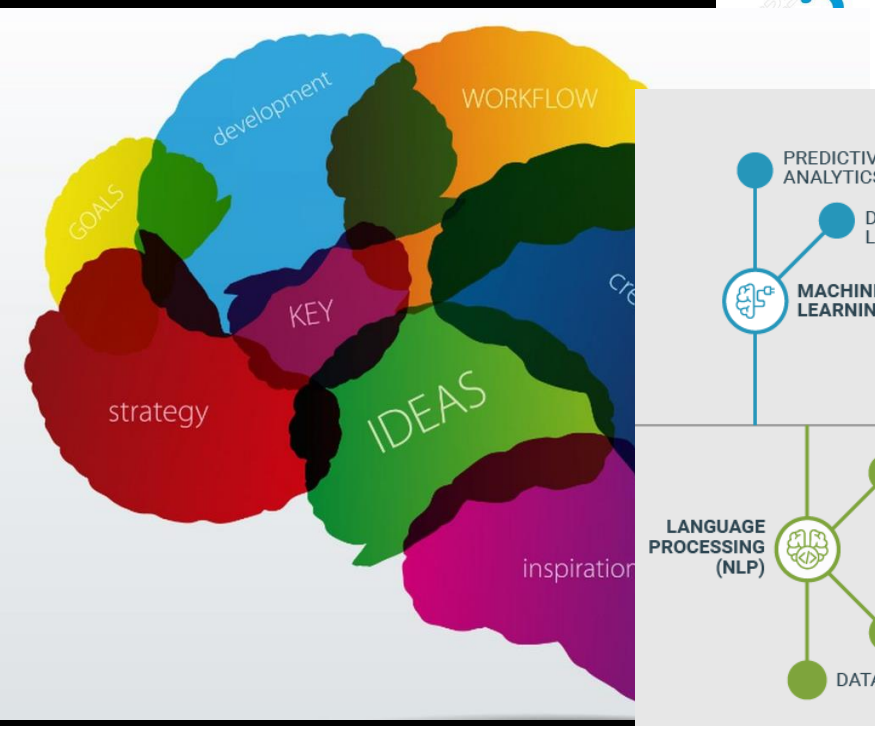


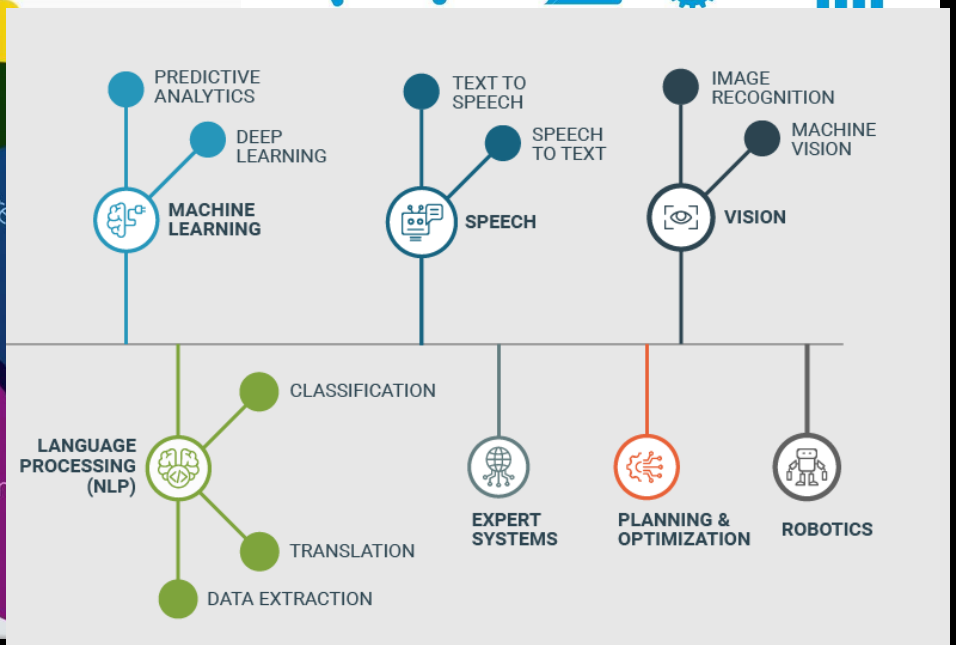





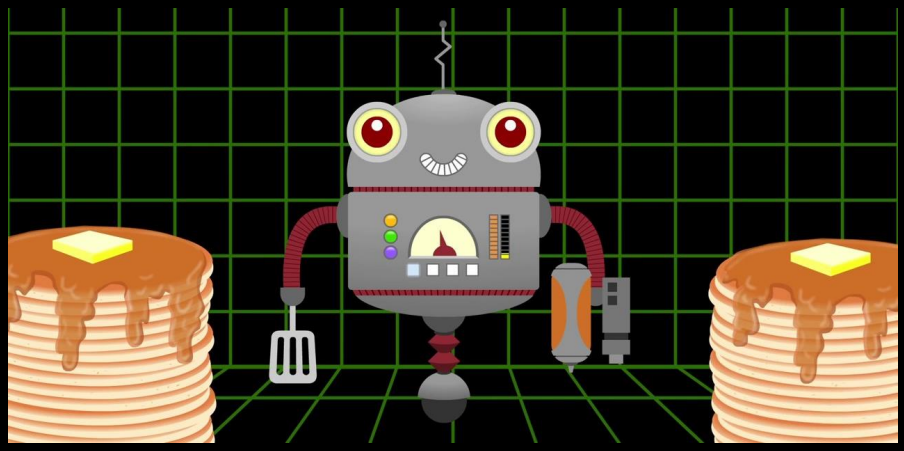


# Intelligence as **cognitive abilities** and **functions**



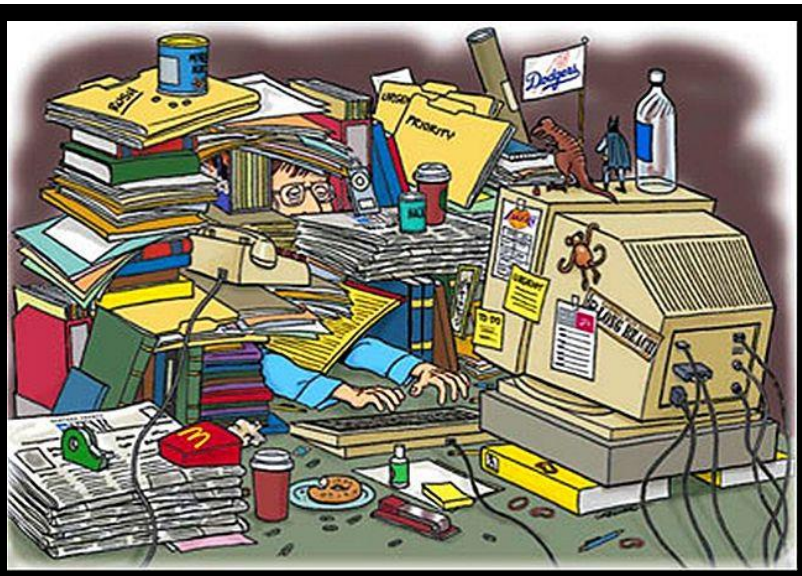


If we want to **build an AI**…



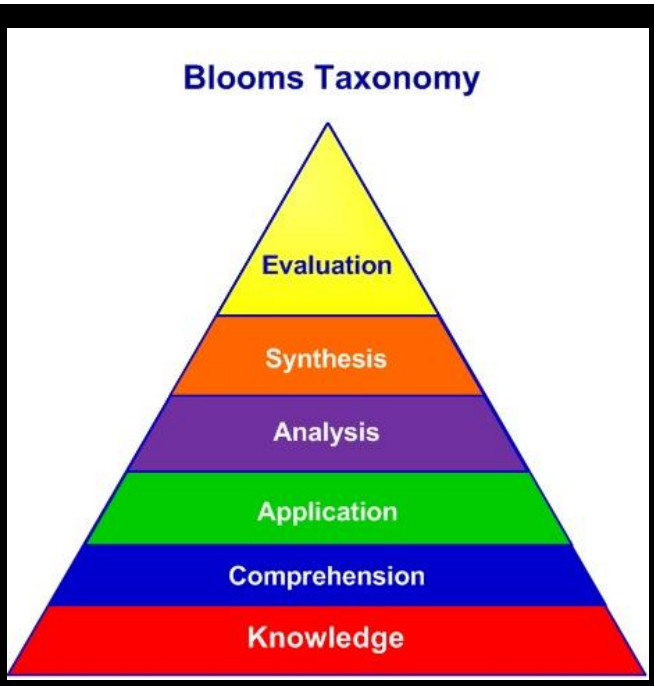
Where do we **start**?

It is a **MESS**!



Today, we **organize the space**

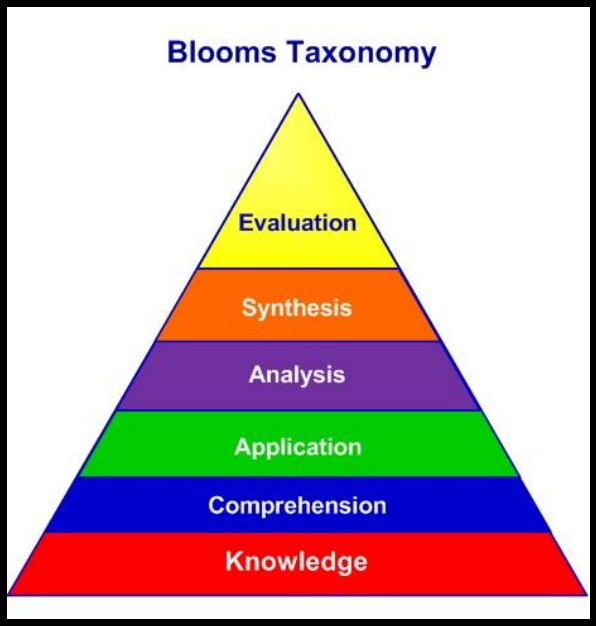
# Fundamental problems of artificial intelligence




Real-world skill tree

# Intended learning outcomes

At the end of the session, you should be able to:




**Define** AI and classic philosophical problems affiliated to AI

**Relate** the key concepts of **decision problems** and **AI systems,** a **framework** for using AI

**Apply** this framework for assessing existing decision problems and AI solutions

**Synthesize** relevant AI solutions and decision problems, given a general domain

## Research shows that learning happens when:

- You are *active* during the learning session.
- You *reflect* over what is learned.
- You put new knowledge in *context*.
- You *connect* new knowledge to previous one.
- You *embrace difficulties*.
- You *do **not** get distracted* by mobile phone and other gadgets.

## What's next?

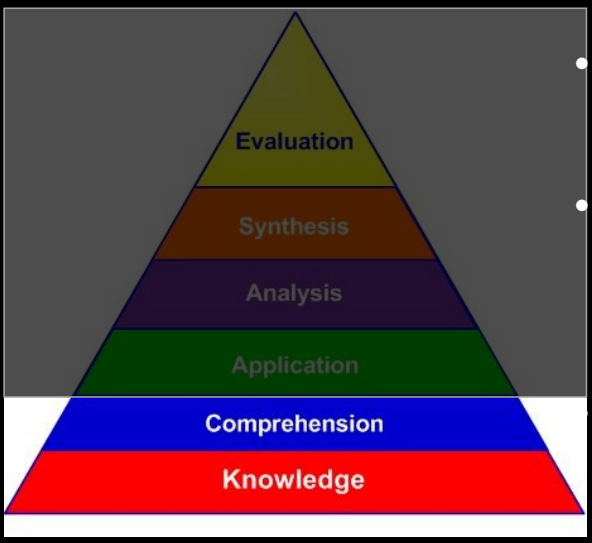
- Three parts (A, B and C) to be prepared
- For each part
  - **For 15 minutes**
    - Breakout rooms, random allocations
    - Discuss the points related to this part you had issues solving
    - Or present your solutions/answers
    - Start with the easiest question for getting moving
    - No more than 5 minutes per question
    - Return to the main class
  - **For 10 minutes:** Break and questions

No chitchat,
Be effective

One more concrete B assignment

- Share a google drive “draw” folder within the group
- Use it as a **big white board**
  collaborative public drawing space
  - Everyone adds missing concepts or links
  - Copy paste your answers if you see some important differences

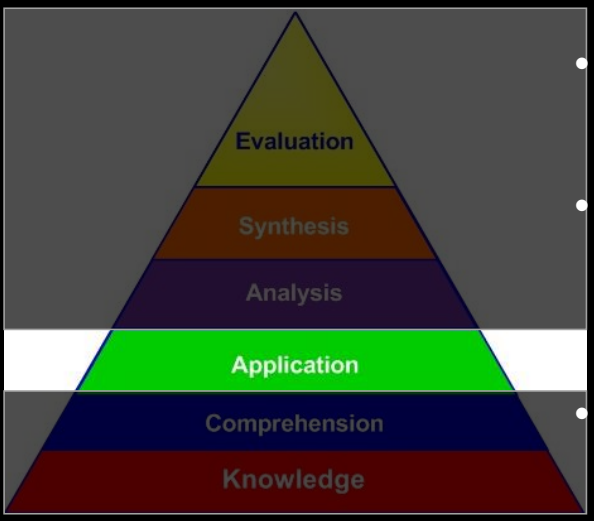
# Part A




- **Define** AI and classic philosophical problems affiliated to AI
- **Relate** the key concepts of **decision problems** and **AI systems,** a **framework** for using AI
- **Apply** these concepts for assessing an existing decision problems and AI solutions
- **Synthesize** new decision problems and relevant AI solution given a general domain

https://tinyurl.com/fundOfAI

# Second activity: Application



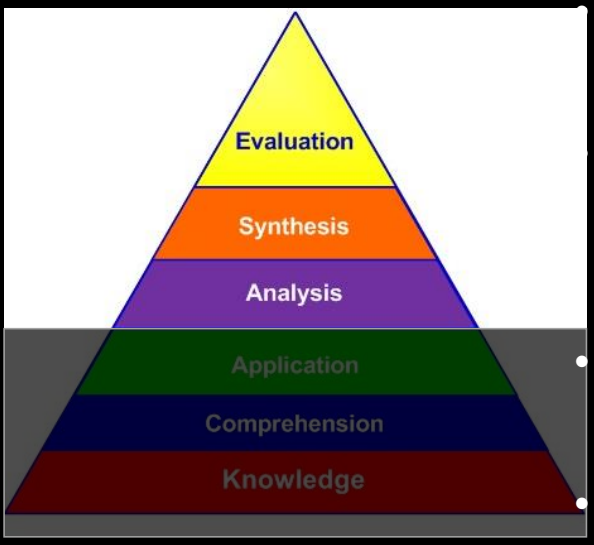

- **Define** AI and classic philosophical problems affiliated to AI
- **Relate** the key concepts of **decision problems** and **AI systems,** a **framework** for using AI
- **Apply** these concepts for assessing an existing decision problems and AI solutions
- **Synthesize** new decision problems and relevant AI solution given a general domain

Instructions

- In turn **each person in the group**
  - Presents one of the subproblems he/she has identified
    - Adds it on the drawing board
  - Presents the cognitive ability for solving this problem input and output
    - Adds it on the drawing board
- **Everyone**: is free to agree or explain why you would do different
- **Duration:** 15 minutes

https://tinyurl.com/fundOfAI

# Third activity: Synthesize



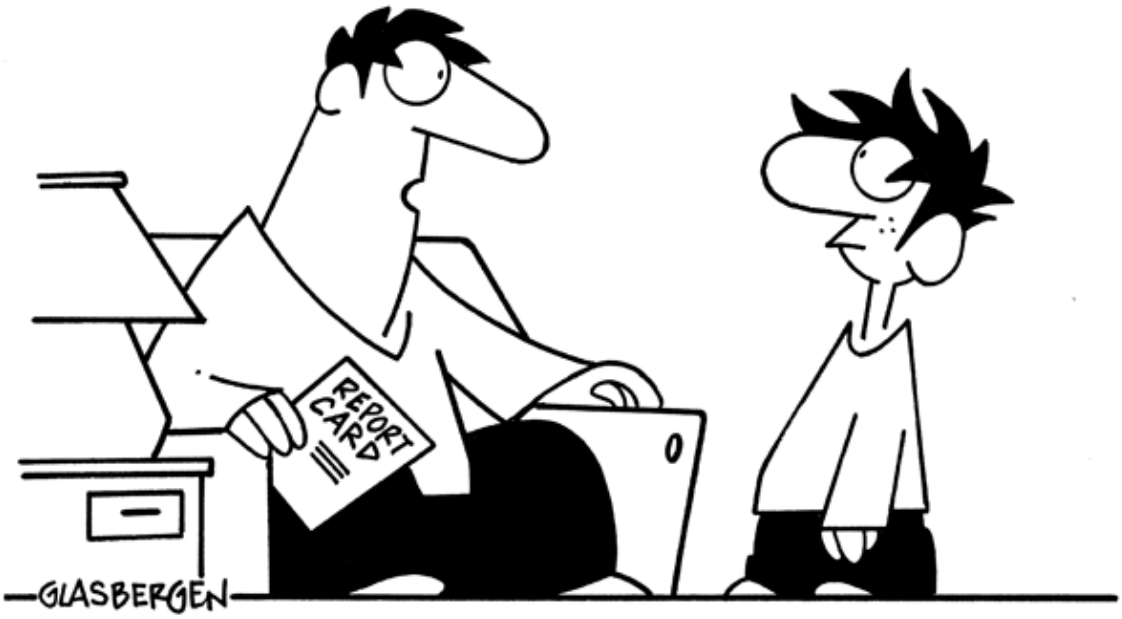

- **Define** AI and classic philosophical problems affiliated to AI
- **Relate** the key concepts of **decision problems** and **AI systems,** a **framework** for using AI
- **Apply** these concepts for assessing an existing decision problems and AI solutions
- **Synthesize** new decision problems and relevant AI solution given a general domain

Instructions: same as previously

- In turn **each person in the group**
  - Presents one of the subproblems he/she has identified
    - Adds it on the drawing board
  - Presents the cognitive ability for solving this problem input and output
    - Adds it on the drawing board
- **Everyone**: is free to agree or explain why you would do different
- **Duration:** 15 minutes

https://tinyurl.com/fundOfAI

© Randy Glasbergen
www.glasbergen.com




**"I probably remember 20% of the stuff I learned in school and forgot the other 90%."**

# Take home message

- **Intelligence** combines of multiple cognitive abilities that are **specialized** for **concrete functions**
- **Designing AI systems** is the art of relating environments, decision problems, cognitive abilities, and desired behavior
- Problems are often interleaved: multiple decision abilities are to be combined
- Deciding what to consider and what to ignore is important for designing useful AI systems
- One can become good at recognizing such structures!
  (without even needing to know how to code ☺)

# If you want to learn more

- Checkout online resources on Canvas
- Mingle with your colleagues
  - Go talk to your colleagues
  - Checkout their report
- Look at existing AI systems
- Sketch how to build your own for a specific problem

Anchoring

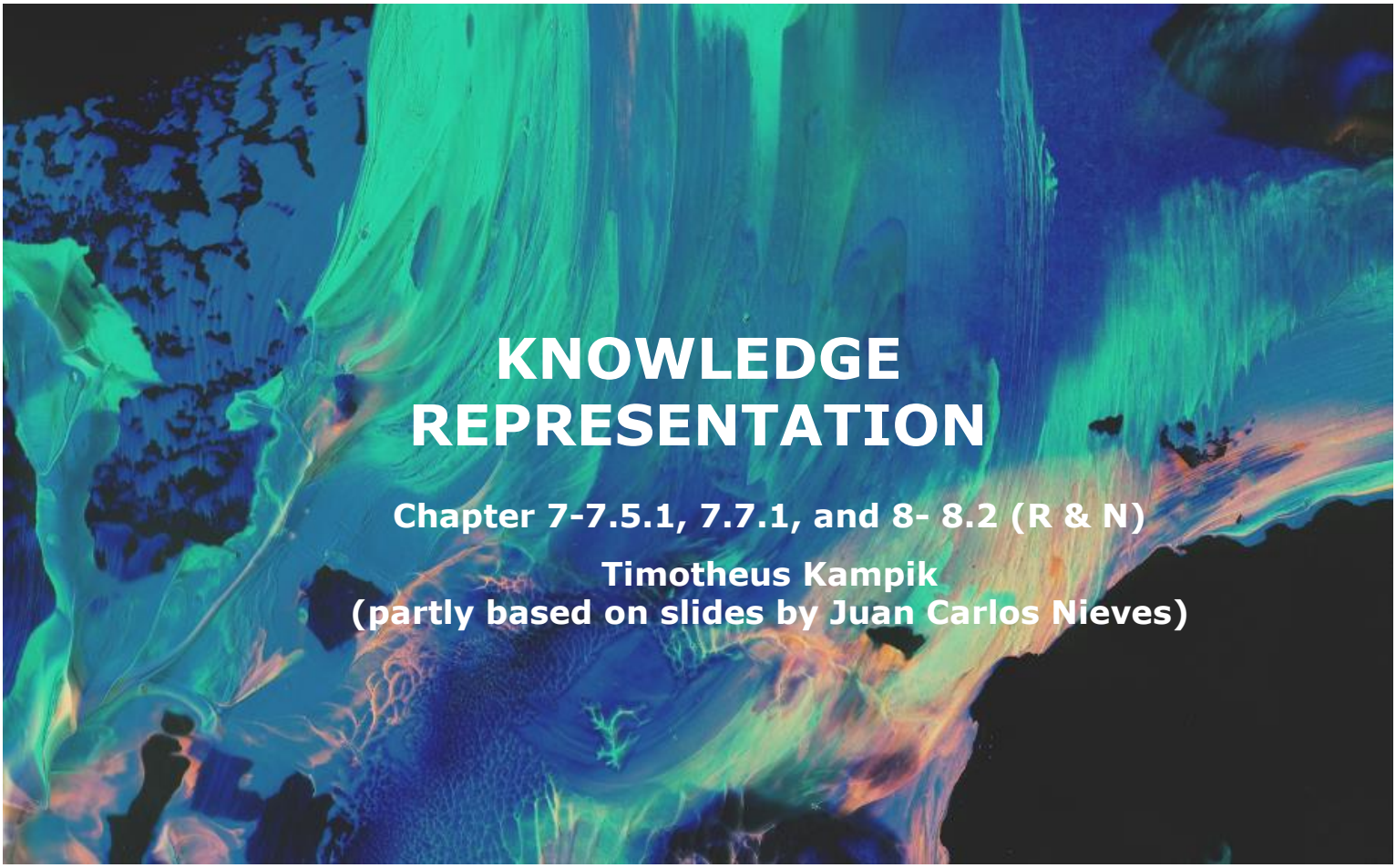
Fill in the following form:

This is anchoring:

State what you learned

Feedback will come later

**https://tinyurl.com/fundOfAI-LR**



# KNOWLEDGE REPRESENTATION

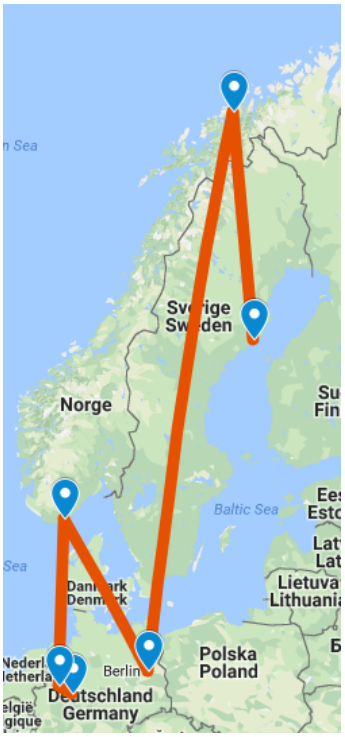
**Chapter 7-7.5.1, 7.7.1, and 8- 8.2 (R & N)**

**Timotheus Kampik**
**(partly based on slides by Juan Carlos Nieves)**

UMEÅ UNIVERSITY

# THAT'S ME

- Bachelor's in Information Systems (Kristiansand)
- Master's Computer Science (Tromsø)
- Currently pursuing PhD at Umeå University:
  - WASP Sweden Autonomous Systems
- Industry background in
  - enterprise systems, product management
  - open source software
- Research interests:
  - AI and decision-making
  - Artificial intelligent agents of bounded rationality
  - Engineering multi-agent systems



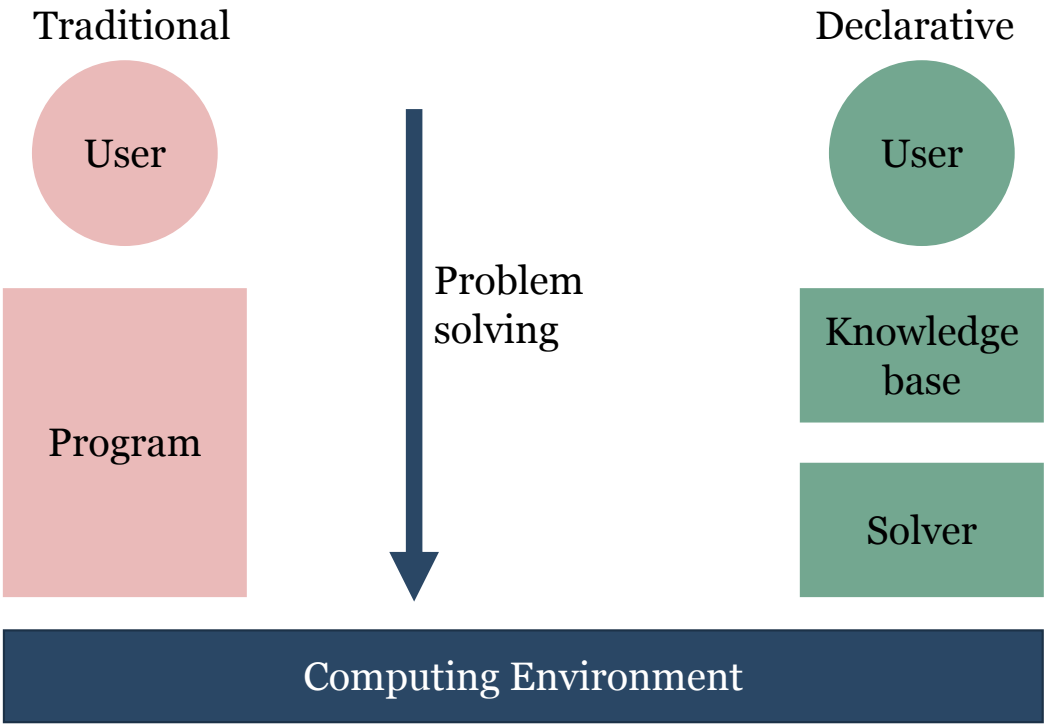
# EXPECTED LEARNING OUTCOMES

- Recap: first-order logic
- Understand declarative programming and difference to “traditional” (imperative) programming
- Understand the difference between monotonic and non-monotonic reasoning
- Gain an intuition of how knowledge representation & reasoning can be applied in practice!

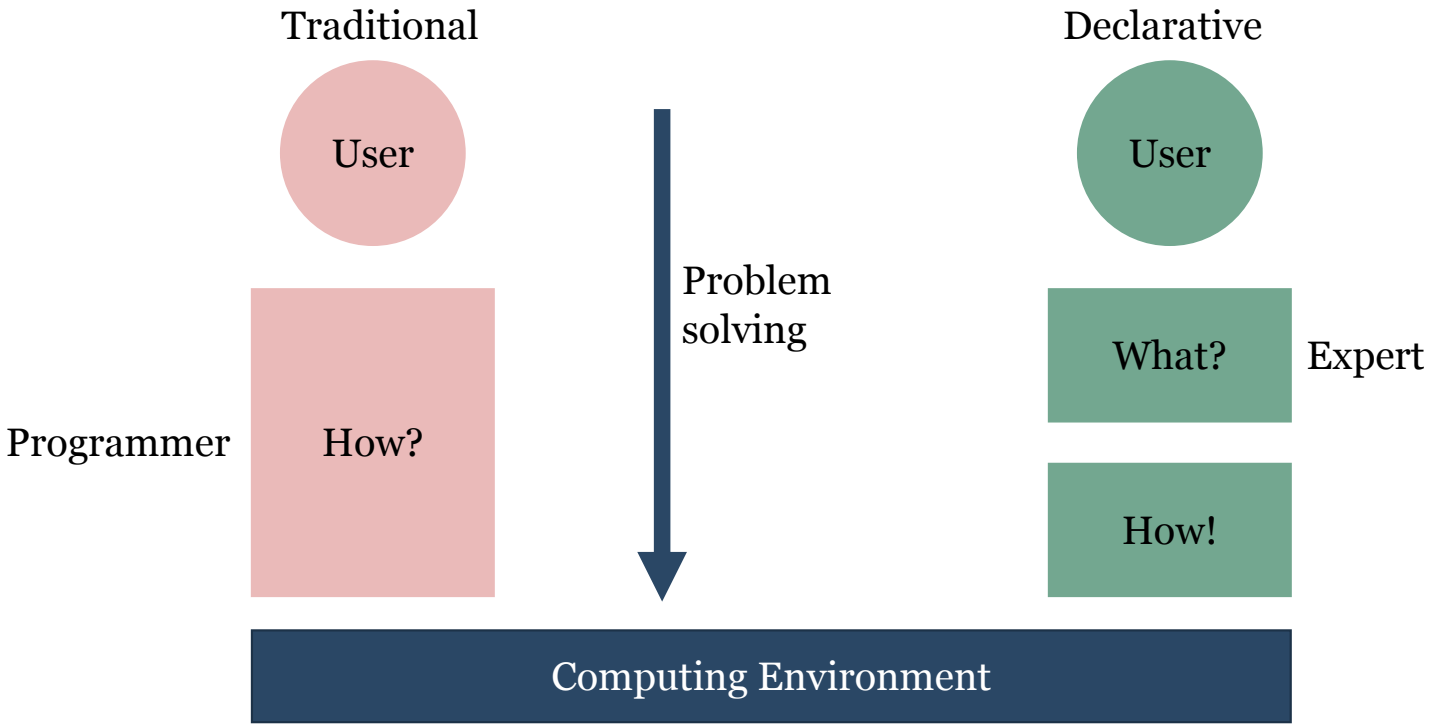
# AGENDA

- Problem solving and knowledge-driven software
- Intuition: use cases
- Logic & First-Order Logic (FOL)
- Monotonic & non-monotonic reasoning
- Applications of knowledge representation & reasoning
  - Monitoring ML-based decision-making
  - First order logics and software development
  - Clinical decision-support

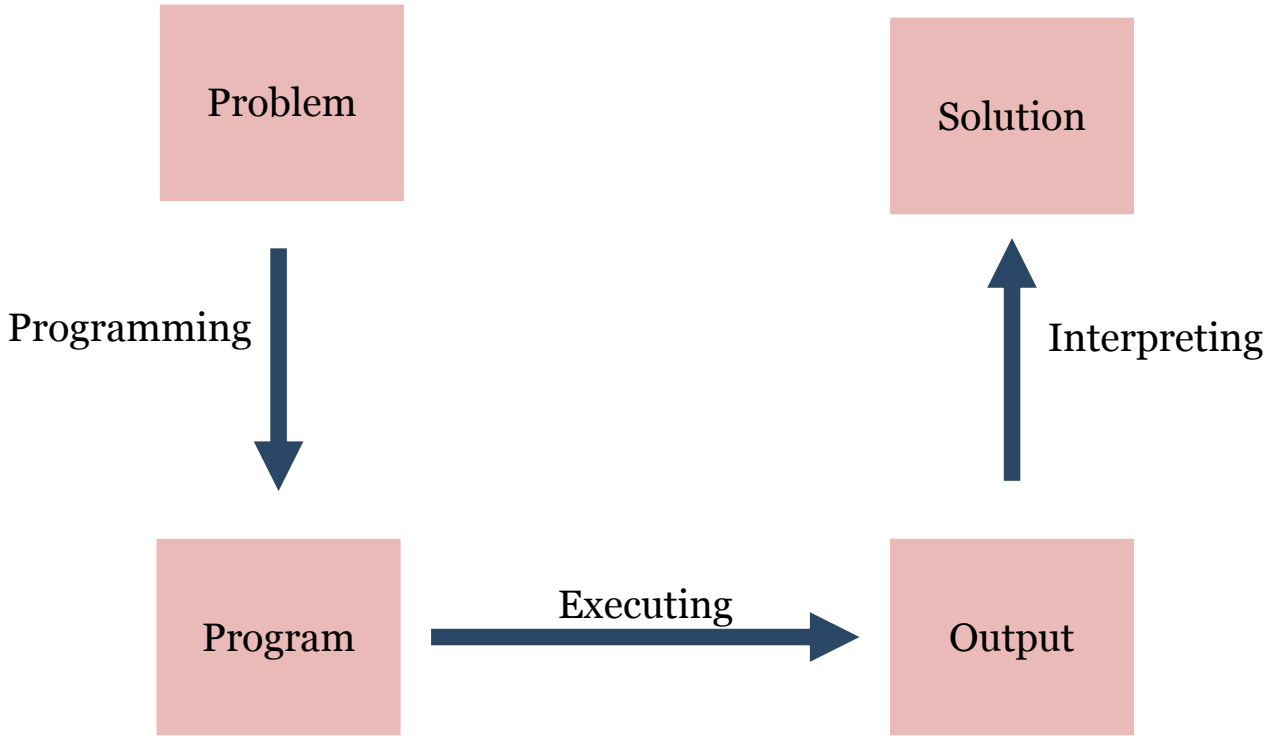
# PROBLEM SOLVING & KNOWLEDGE DRIVEN SOFTWARE



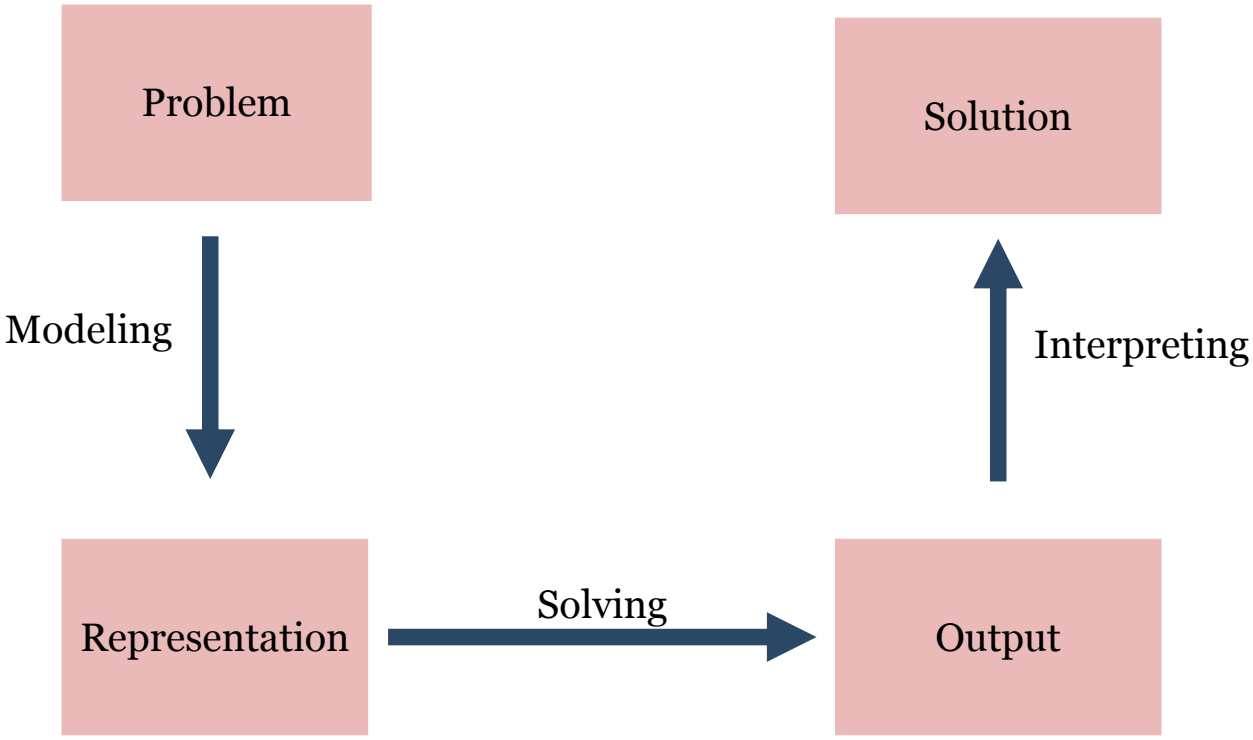
# PROBLEM SOLVING & KNOWLEDGE DRIVEN SOFTWARE



# TRADITIONAL PROGRAMMING



# DECLARATIVE PROGRAMMING

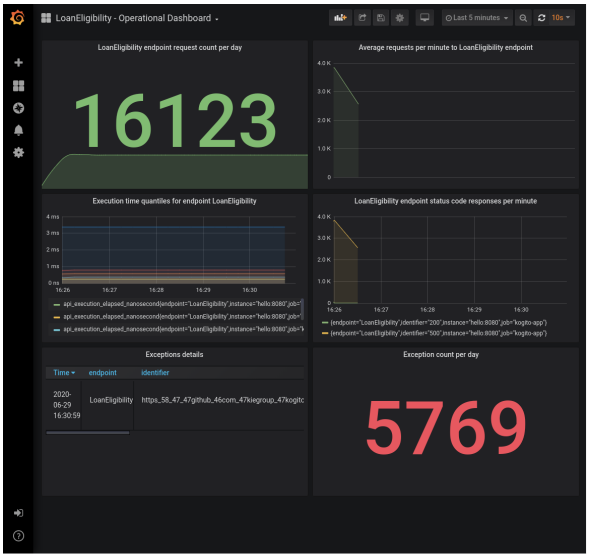


# MOTIVATION: WHY KNOWLEDGE REPRESENTATION & REASONING?

## Use Cases?

# MOTIVATION: WHY DECLARATIVE PROGRAMMING?

Decision Monitoring




https://blog.kie.org/2020/06/trusty-ai-introduction.html
Rebecca Whitworth: ”TrustyAI introduction”
https://blog.kie.org/2020/07/trustyai-meets-kogito-decision-monitoring.html
Jacopo Rota. ”TrustyAI meets Kogito: decision monitoring

Clinical Decision-Support

https://www.redhat.com/files/resources/mi-rtinsights-northwell-analyst-paper-f12456-201805-en.pdf
redhat: ”Building a clinical decision engine platform with Red Hat Decision Manager

# Reflect: when to use traditional programming, when declarative programming?

# LOGICS AND FIRST-ORDER LOGIC

# LOGICS

- Logics are **formal languages** for representing **information** so that **conclusions** can be drawn
- A logics has:
  - **Syntax**: defines the sentences in the language/**grammar**
  - **Semantics**: defines the **meaning** of sentences or the truth of a sentence in a world

# LOGICS

Example in arithmetic:

- `x + 2 > y` is a sentence
- `x + 2 >` is not a sentence
- `x + 2 > y` is true in a world where `x = 2, y = 1`
- `x + 2 > y` is false in a world where `x = 2, y = 8`

# EXERCISE - LOGICS

Exercise, examples in arithmetic:

- Is `-(x + 2) > y` a sentence?
- Is `i++ == 2` a sentence?
- Is `i++ == 2` a sentence true in a world where i = 1?

# LANDSCAPE OF LOGIC-BASED LANGUAGES: MONOTONIC VS. NON-MONOTONIC

- Monotonic: the truth of a proposition or axiom does not change when new information is added
  `All humans are mortal AND Socrates is a human ➔ Socrates is mortal`
- Non-monotonic: the truth of a proposition or axiom may change when new information is added
  `Birds typically fly AND Tweety is a bird ➔ Tweety presumably flies`

# OPEN QUESTION: HOW TO ”REASONABLY” RESTRICT NON-MONOTONIC REASONING?

- See: *Gabbay. Theoretical Foundations for Non-Monotonic Reasoning in Expert Systems*
- Although the paper is from 1985, the question remains open

# QUESTION TO YOU: CAN WE REASON NON-MONOTONICALLY WITH SIMPLE IF-CONDITIONS?

Example, JavaScript:

```
const rule = kb => {
     if (kb.isRaining === true)
     console.log("take umbrella")
}
```

# LANDSCAPE OF LOGIC-BASED LANGUAGES

- Predicate logics (first order logic and its sub-languages)
  - Description logic
  - Web ontology logic
  - Extensions of FOL: Modal and epistemic logics
- Inconsistency-tolerant logic (non-monotonic)
- Representing vagueness
  - Probabilistic logics
  - Bayesian networks
  - Markov chains

# FIRST-ORDER LOGIC (FOL)

- First-Order Logic (or: predicate logic) as a classical **monotonic** language is composed of
  1. Different types of **symbols**:
     - **Constants** or individuals of the world (upper case), e.g.:
       `John, 2, B Apple, Red, 9, Sara`
     - **Functions** that map individuals to individuals, e.g.:
       `motherOf(John) = Sara`
     - **Predicates** that map individuals to truth values, e.g.:
       `smaller(2, 9), color(Apple, Red)`

# FIRST-ORDER LOGIC (FOL)

- First-Order Logic (or: predicate logic) as a classical **monotonic** language is composed of
  2. A set of **primitives**:
     - **Variables** (lower case), e.g.:
       `x, y, z`
     - **Connectives** (or logical operators):
       `¬, ∧,∨,⇔,⇒,(,)`
       **Quantifiers**:
       Universal (∀): ∀x, P: holds for all instantiations of P, e.g.:
       `∀x, (mother(x) ⇒ parent(x))`
       Existential (∃): ∃x, P: holds for some instantiations of P, e.g.:
       `∃x, (man(x) ∧ human(x))`

# FIRST-ORDER LOGIC (FOL)

- First-Order Logic (or: predicate logic) as a classical **monotonic** language is composed of
  3. **Terms**:
     - Variables
     - Constants
     - n-ary functions applied to n arguments, e.g.:
       `a, x, f(a), g(x, b), f(g(x, g(b)))`
     - 0-ary function is a constant

# FIRST-ORDER LOGIC (FOL)

- First-Order Logic (or: predicate logic) as a classical **monotonic** language is composed of
  4. **Atoms**: an n-ary predicate applied to n terms, e.g.: p(a)
  5. **Literals**: atoms or their negation, e.g.: `p(f(x), g(x, f(x))),¬p(f(x),¬g(x, f(x)))`
  6. **Formula** (or sentences):
     - Literal
     - Application of logical connectives to formula, e.g.: `p(f(x), x)⇒p(f(g(x, y))∧g(x, y))`

# FOL SEMANTICS – TRUTH VALUES

- To find a meaning for a sentence (or formula) we need to calculate its truth value
- Each formula is evaluated to 2 truth values: true or false
- The truth value of a sentence depends on the truth values of the symbols in the sentence
- Determining the truth values of a formula is called interpretation
- Truth table enumeration is one way to calculate the truth value of a formula

# FOL SEMANTICS – TRUTH VALUES

Truth table of the two symbol A and B, for all logical connectives:

| A | B | ¬ A | A ∧ B | A ∨ B | A ⇒ B | A ⇔ B |
|---|---|---|---|---|---|---|
| true | true | false | true | true | true | true |
| true | false | false | false | true | false | false |
| false | true | true | false | true | true | false |
| false | false | true | false | false | true | true |

Given the “A” and “B” columns, you should be able to **compute (not memorize!)** the rest of the content of the table.

# FOL SEMANTICS – INTERPRETATION

- An interpretation is an **assignment of meaning** to the symbols of a language.

- Logical formula that are syntactically correct do not have any meaning until they are given some interpretation.

- If a given interpretation assigns the value true to a formula, the interpretation is called a model of that formula.

# FOL SEMANTICS – INTERPRETATION

An interpretation I: `(D,α)` consists of:

- Domain `D`: non-empty set of values
- Function α that assigns:
  - To each variable x, a value α[x] ∈ D
  - To each n-ary function f, a value in D:
    $$\alpha[f] : D^n \rightarrow D$$
  - To each n-ary predicate P, either true or false:
    $$\alpha[P] : D^n \rightarrow \{ true, false \}$$

# FOL SEMANTICS – EXAMPLE I

$$F: q(x, y) \wedge p(x) \Rightarrow \neg p(y)$$

Interpretation $I$: $(D, \alpha)$, where:

- Domain $D = \{john = Father, sara = Mother, kate = Daughter, bob = Son\}$
- Function α:
  - for variables x and y :α[x] = john, α[y] = sara
  - for predicates p and q :-
    unary predicateα[p]: D1 → { true, false }
    p(x) returns true if x is male, otherwise false
    binary predicateα[q]: D2→{ true, false }
    q(x,y) returns true if x is sibling of y, otherwise false

The given formula F is true under I, and therefore I is a model of F

# FOL SEMANTICS – EXAMPLE II

$$F: q(x, y) \wedge p(x) \Rightarrow \neg p(y)$$

Interpretation $I$: $(D, \alpha)$, where:

- Domain $D = \{john = Father, sara = Mother, tom = Son, bob = Son\}$
- Function α:
  - for variables x and y :α[x] = tom, α[y] = bob
  - for predicates p and q :-
    unary predicate α[p]: D1 → { true, false }
    p(x) returns true if x is male, otherwise false
    binary predicate α[q]: D2→{ true, false }
    q(x,y) returns true if x is sibling of y, otherwise false

The given formula F is false under I, and therefore I is not a model of F

# FOL SEMANTICS – FORMULA

A formula is called:

- **Satisfiable** iff it is true for at least one interpretation. Therefore, the previous formula is satisfiable!
- **Valid** iff it is true for all interpretations
- **Unsatisfiable** iff it is not true for any interpretation

- Every interpretation that satisfies a formula is called a model of the formula

# KNOWLEDGE BASE

A Knowledge Base (KB) is a set of sentences (formula) that are all connected with conjunction (∧), describing a situation in which the world is in.

Example:

- `Σ={ R=it is raining, W=streets are wet, D=it is dark, C=it is cold, H=John is home }`
- Then, a KB can be defined as:
  `KB ={ (R⇒W) ∧ (D∧C⇒H) ∧ (D) ∧ (¬H) }`
  translated to:
  `If it is raining then the streets are wet.`
  `If it is dark and cold then John is at home.`
  `It is dark.`
  `John is not home.`

# ENTAILMENT

Entailment ($\vDash$) means that one thing follows from another:

Example: KB $\vDash \alpha$ means that the knowledge base KB entails the sentence $\alpha$ if $\alpha$ is true in all words in which KB is true.

# EXERCISE

Use a truth table to figure out whether the statement `people are sad` is entailed from this KB:

- `If it is dark and raining, people are sad.`
- `If it is raining, then it is dark.`
- `It is raining.`

# REASONING

# REASONING

- Reasoning or *inference* is the process of answering a question:
  $KB \vdash_p \alpha$ means $\alpha$ can be derived from KB by procedure p.

- Consequences of KB are a haystack, $\alpha$ is a needle
  - **Entailment**: needle is in the haystack
  - **Inference**: finding the needle, e.g. by truth table enumeration

# REASONING

- Formal properties of a reasoning process:
  - **Soundness**: procedure p is sound iff:
    $(KB \vdash_p \alpha) \Rightarrow (KB \vDash \alpha)$
    KB entails everything that p infers
  - **Completeness**: procedure p is complete iff:
    $(KB \vDash \alpha) \Rightarrow (KB \vdash_p \alpha)$
    p infers everything that KB entails

# COMPLETENESS ”IN THE WILD”

- Important for managing “business rules” (decision-support and automation systems)
- Identify missing rules
- Merge overlapping rules
- See: *Calvanese et al. Semantics, Analysis and Simplification of DMN Decision Tables*

# COMPLETENESS ”IN THE WILD”

Decision logic: Check applicant

Decision Table
Expression
Link
Informal Expression

Add rule | Remove rule | Add input | Remove column | Errors detected | Import/Export

| U | Inputs: ...tter of Application | | Inputs: Evaluate overall skills | | Outputs: Status | Annotations: New Annotation |
|---|---|---|---|---|---|---|
| | {complete,incomplete} | | {Acceptable,Unacceptable} | | {Accept,Reject} | Informational |
| 1 | = | complete | = | Acceptable | Accept | The candidate is worthy of an interview |
| 2 | = | incomplete | = | Acceptable | Reject | |
| 3 | = | complete | = | Unacceptable | Reject | |
| + | Add new row | | | | | |

Check Decision Table

| # | Rule | Description |
|---|---|---|
| 1 | --- | No rule exists for ( incomplete, !Acceptable ) |

Cancel | Apply

UMEÅ UNIVERSITY

# REASONING METHODS

Divide into (roughly)

- Model checking
  - Models are assignments of true and false to every symbol
  - Truth table enumeration
- Application of inference rules
  - Legitimate (sound) generation of new sentences from old
  - Proof = a sequence of inference rule applications
  - Typically require translation of sentences into a normal form
- Reasoning can be monotonic or non-monotonic!

# APPLICATIONS

# EXERCISE - APPLICATIONS

Think about some applications of

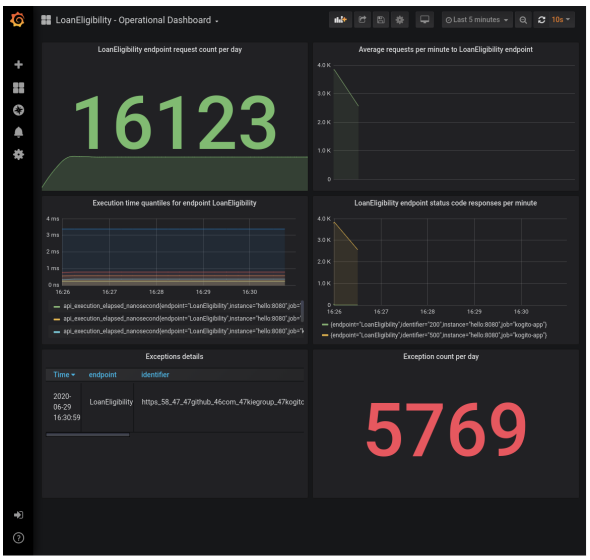
- Declarative programming
- First-order logic
- Non-monotonic reasoning

# DECLARATIVE PROGRAMMING: DECISION MONITORING OF MACHINE LEARNING-BASED SYSTEMS

- Machine learning increasingly used for decision-automation
- Still, compliance with rules and regulations need to be checked or be guaranteed “by design”

➔Use declarative, rule-based systems for compliance checks

➔“Learn” human interpretable rules




https://blog.kie.org/2020/06/trusty-ai-introduction.html
Rebecca Whitworth: ”TrustyAI introduction”
https://blog.kie.org/2020/07/trustyai-meets-kogito-decision-monitoring.html
Jacopo Rota. ”TrustyAI meets Kogito: decision monitoring

# FIRST ORDER LOGIC: RELEVANCE FOR SOFTWARE ENGINEERING

If-conditions and test assertions resemble first order logic

➔Understanding of first order logic a crucial part of a programmer's tool box

```
@[test]
+!test_egreedy_bandit_api :
    true
    <-
    //.init_bandit(tArmNames, type = "EPSILON_GREEDY", epsilon,
    decayRate, seed, handoverArg, unifier)
    .init_bandit([a, b, c, d, e], "EPSILON_GREEDY", 0.1, 0.1, 20, B);
    !assert_equals(0, B);
    .pull_arm(B, A);
    !assert_equals("e", A);
    .set_reward(B, A, 2, F);
    !assert_equals([0, 0, 0, 0, 1], F);
.
```

# NON-MONOTONIC REASONING: RELEVANCE FOR DECISION SUPPORT & AUTOMATION

- Most large organizations use complex, rule-based systems
- Again, rules are necessary for compliance and quality assurance
- Non-monotonic reasoning (AKA: defeasible reasoning) allows for a structured approach towards managing rule conflicts

https://www.redhat.com/files/resources/mi-rtinsights-northwell-analyst-paper-f12456-201805-en.pdf
redhat: "Building a clinical decision engine platform with Red Hat Decision Manager

# RECAP – LEARNING GOALS

# EXPECTED LEARNING OUTCOMES

- **Recap: first-order logic**
- **Understand declarative programming and difference to “traditional” (imperative) programming**
- **Understand the difference between monotonic and non-monotonic reasoning**
- **Gain an intuition of how knowledge representation can be applied in practice!**

# FURTHER READING

- *Russel, Norvig: Artificial Intelligence: A Modern Approach, chapters* 7-7.5.1, 7.7.1, and 8-8.2

*Plus (optional, but recommended to take a look):*

- *Gabbay. Theoretical Foundations for Non-Monotonic Reasoning in Expert Systems*
- *Calvanese et al. Semantics, Analysis and Simplification of DMN Decision Tables*

# FURTHER CODING

- *Drools, industry-scale rules engine for declarative programming:* https://docs.jboss.org/drools/release/5.3.0.Final/drools-expert-docs/html/ch01.html

UMEÅ UNIVERSITY

# FUNDAMENTALS OF ARTIFICIAL INTELLIGENCE

## Lecture 6: Search and planning

Ola Ringdahl

UMEÅ UNIVERSITY

# SEARCH

## R&N Chapter 3

UMEÅ UNIVERSITY

# PROBLEM SOLVING

- Rational agents need to perform sequences of actions in order to achieve goals.
- Possible to use look-up table telling the agent what to do in every circumstance, but:
  - Difficult to build and all contingencies must be anticipated
- More general approach: agent has knowledge of the world and how its actions affect it
- This is the general task of **problem solving** and is typically performed by **searching** through an internally modelled space of world states

# PROBLEM SOLVING TASK

- Given:
  - **An initial state** of the world
  - A set of possible actions or **operators** that can be performed.
  - A **goal test** that can be applied to a single state of the world to determine if it is a goal state.
- Find:
  - A **solution** in form of a **path** of states and operators that shows how to transform the initial state into one that satisfies the goal test.
  - The initial state and set of operators implicitly define a **state space** - *the set of all states reachable from the initial state by any sequence of actions*

# MEASURING PERFORMANCE

- **Path cost**: a function that assigns a cost to a path, typically by summing the cost of the individual operators in the path. Usually want to find minimum cost solution (*optimal*).
- **Search cost**: The computational time and space (memory) required to find the solution.
- Usually a trade-off between path cost and search cost
  - Must find the best solution in the time that is available.

# SEARCH STRATEGIES

- Easiest way to implement various search strategies is to maintain a *queue* of unexpanded search nodes.
- Different strategies result from different methods for inserting new nodes in the queue

```
function GENERAL-SEARCH( problem, QUEUING-FN) returns a solution, or failure

  nodes ← MAKE-QUEUE(MAKE-NODE(INITIAL-STATE[problem]))
  loop do
     if nodes is empty then return failure
     node ← REMOVE-FRONT(nodes)
     if GOAL-TEST[problem] applied to STATE(node) succeeds then return node
     nodes ← QUEUING-FN(nodes, EXPAND(node, OPERATORS[problem]))
  end
```

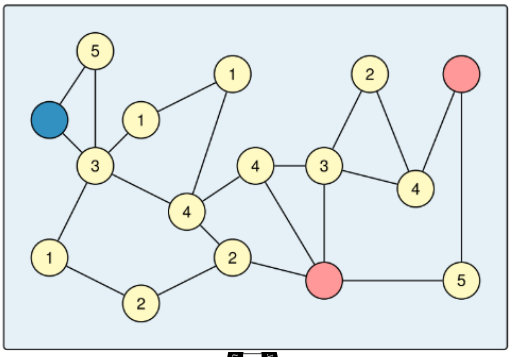
# SEARCH STRATEGIES

Properties of a search strategy:

- **Completeness** - guaranteed to find a solution if there is one?
- **Optimality** - does it find the best solution?
- **Time complexity** - how long time to find a solution?
- **Space complexity** - how much memory is required during the search?

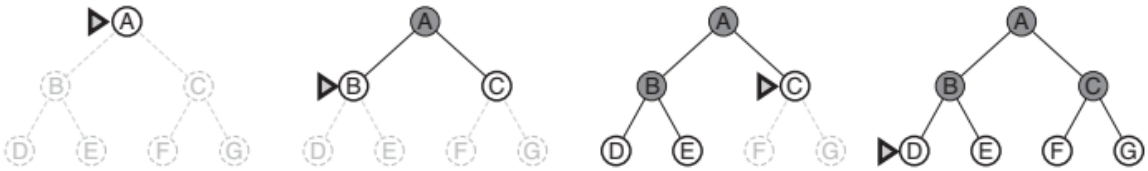
# SEARCH ALGORITHMS

| Uninformed search algorithms | Informed search algorithms |
|---|---|
| • Depth first<br>• Breadth first<br>• Uniform cost<br>• Depth limited<br>• Iterative deepening<br>• Bidirectional | • Greedy best first<br>• A*<br>• Memory-bounded heuristic |



# BREADTH FIRST SEARCH

- Expands search nodes level by level, all nodes at level *d* are expanded before expanding nodes at level *d*+1




- Implemented by adding new nodes to the end of the queue (FIFO queue):

  GENERAL-SEARCH(problem, ENQUEUE-AT-END)

- **Complete** - eventually visits every node to a given depth
- **Optimal** - provided path cost is a nondecreasing function of the depth of the node (e.g. all actions have same cost) since nodes explored in depth order.

# BREADTH FIRST COMPLEXITY

- Assume there are an average of $b$ successors to each node, called the **branching factor**.
- To find a solution path of length $d$ we must explore

$$1 + b + b^2 + b^3 + \cdots + b^d = O(bd)$$

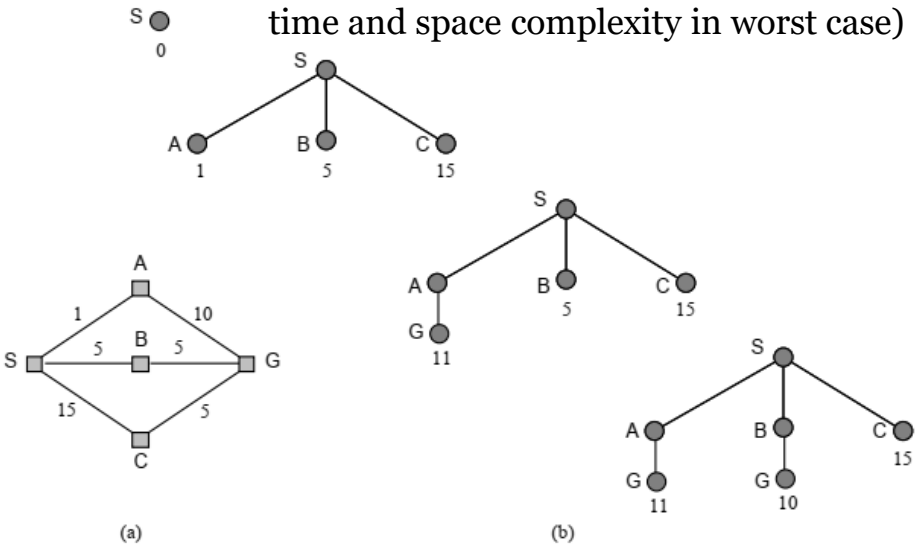
- Need $b^d$ nodes in memory to store leaves in the queue

| Depth | Nodes | Time | Memory |
|---|---|---|---|
| 2 | 110 | .11 milliseconds | 107 kilobytes |
| 4 | 11,110 | 11 milliseconds | 10.6 megabytes |
| 6 | $10^6$ | 1.1 seconds | 1 gigabyte |
| 8 | $10^8$ | 2 minutes | 103 gigabytes |
| 10 | $10^{10}$ | 3 hours | 10 terabytes |
| 12 | $10^{12}$ | 13 days | 1 petabyte |
| 14 | $10^{14}$ | 3.5 years | 99 petabytes |
| 16 | $10^{16}$ | 350 years | 10 exabytes |

**Figure 3.13** Time and memory requirements for breadth-first search. The numbers shown assume branching factor $b = 10$; 1 million nodes/second; 1000 bytes/node.

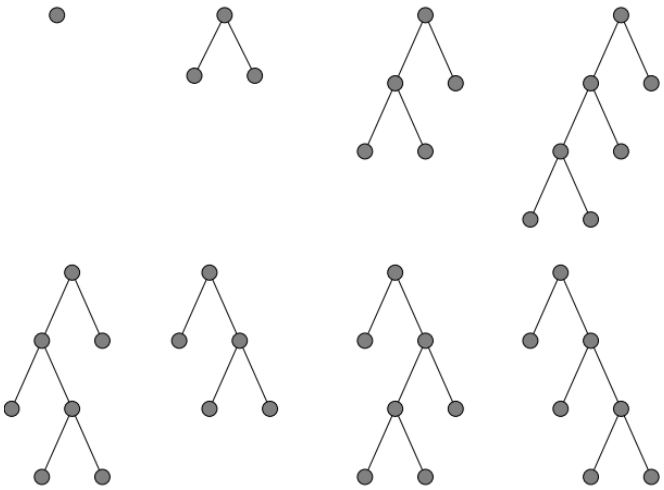
# UNIFORM COST SEARCH

- Like breadth-first except **always expand node of lowest path cost** instead of least depth (i.e. sort new queue by path cost).
- Do not recognize goal until it is the least-cost node on the queue (i.e. when the goal node is selected for expansion).
  - Therefore, **optimal** with non-negative step costs (also means higher time and space complexity in worst case)



# DEPTH FIRST SEARCH

- Always expand node at deepest level of the tree, i.e. one of the most recently generated nodes. When hit a dead-end, backtrack to last choice.



- Implemented by adding new nodes to front of the queue (LIFO): GENERAL-SEARCH(problem, ENQUEUE-AT-FRONT)

# DEPTH-FIRST PROPERTIES

- **Not complete** - might get lost following infinite path.
- **Not optimal -** can find deeper solution before shallower ones explored.
- Worst case **time complexity $O(b^d)$** - need to explore entire tree.
  - May find a solution quickly before exploring the whole space.
- **Space complexity** is **$O(b^m)$** where m is maximum depth of the tree
  - The queue just contains a single path from the root to a leaf node along with remaining sibling nodes for each node along the path.
- Can add a **depth limit** $l$ to prevent exploring nodes beyond a given depth.
  - Prevents infinite path. Still **incomplete** if no solution within depth limit

# ITERATIVE DEEPENING

- Conduct a series of depth-limited searches, increasing depth-limit each time.

```
function ITERATIVE-DEEPENING-SEARCH(problem) returns a solution, or failure
  for depth = 0 to ∞ do
    result ← DEPTH-LIMITED-SEARCH(problem, depth)
    if result ≠ cutoff then return result
```

- Most nodes are at the bottom level, so only adds constant time
- Depth-first:

$$1 + b + b^2 + \cdots + b^{d-2} + b^{d-1} + b^d$$

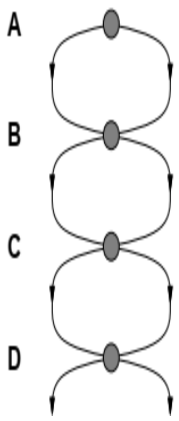
- Iterative deepening:

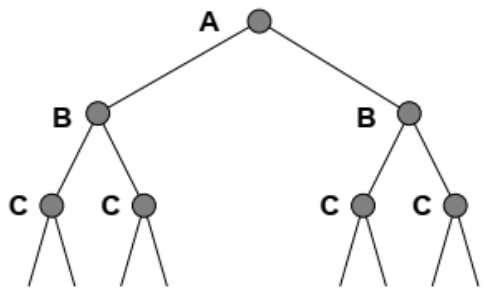
$$(d+1)1 + db + (d-1)b^2 + \cdots + 3b^{d-2} + 2b^{d-1} + 1b^d$$

- **Time complexity** is still **O(*b*$^d$)**
- **Space complexity O(*b*$^m$)**

# AVOIDING REPEATED STATES

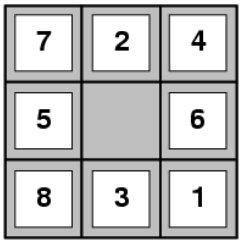
- Basic search methods may repeatedly search the same state if it can be reached via multiple paths.
- Three methods for reducing repeated work in order of effectiveness and computational overhead:
  - Do not follow self-loops (remove successors back to the same state).
  - Do no create paths with cycles (remove successors already on the path back to the root). O(*d*) overhead.
  - Do not generate any state that was already generated. Requires storing all generated states (O($b^d$) space) and searching them (usually using a hash-table for efficiency).



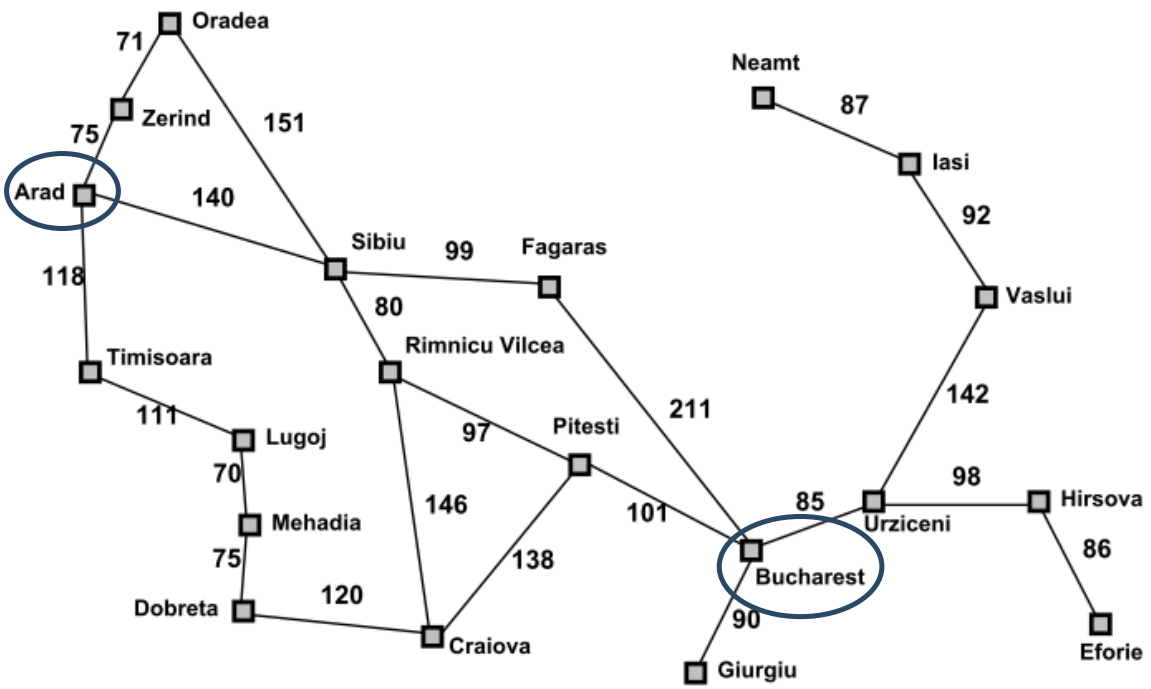



# INFORMED SEARCH

- Use a heuristic function to estimate the cost of getting from the current state to a goal state.
- Reasonable heuristic functions can only be constructed using domain specific knowledge; i.e. they need to be informed.
- Examples:
  - When searching on a road map, use straight line distance.
  - Eight puzzle: no. of misplaced tiles
  - Sorting: how many pairs are in incorrect order?
- Note: these are not necessarily good heuristic functions.



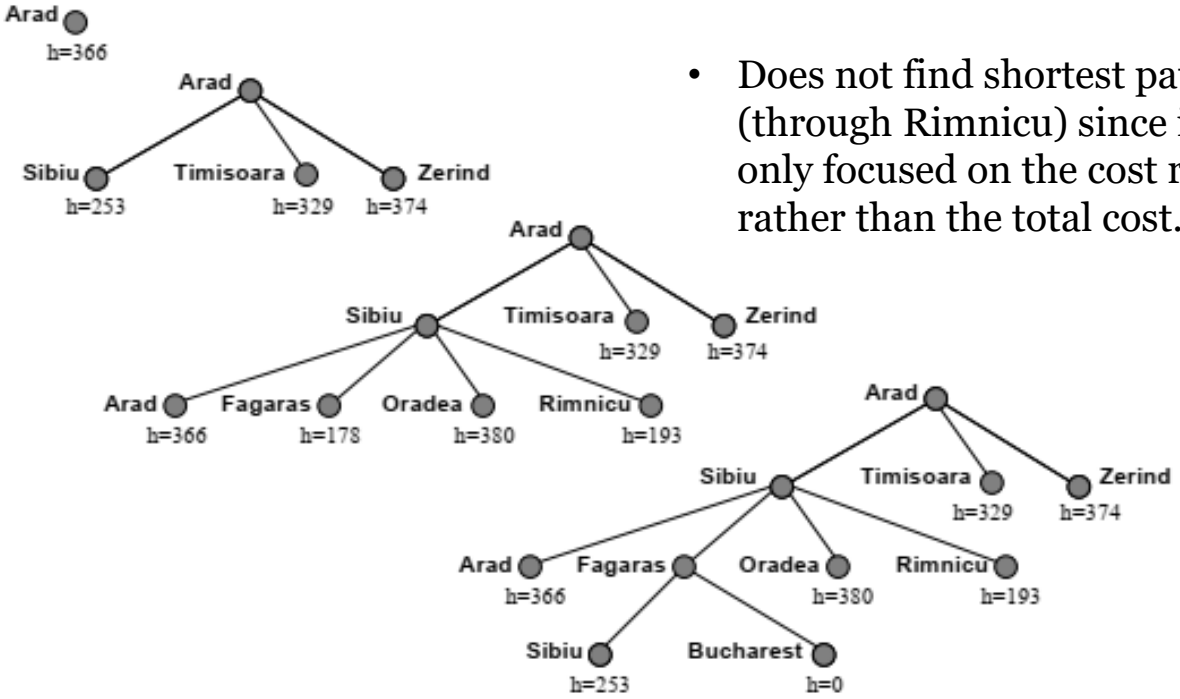
# ROMANIA WITH STEP COSTS IN KM




| Straight−line distance to Bucharest | |
|---|---|
| Arad | 366 |
| Bucharest | 0 |
| Craiova | 160 |
| Dobreta | 242 |
| Eforie | 161 |
| Fagaras | 178 |
| Giurgiu | 77 |
| Hirsova | 151 |
| Iasi | 226 |
| Lugoj | 244 |
| Mehadia | 241 |
| Neamt | 234 |
| Oradea | 380 |
| Pitesti | 98 |
| Rimnicu Vilcea | 193 |
| Sibiu | 253 |
| Timisoara | 329 |
| Urziceni | 80 |
| Vaslui | 199 |
| Zerind | 374 |

# GREEDY BEST-FIRST SEARCH

- Tries to expand the node closest to the goal – hoping that it leads to a solution quickly




- Does not find shortest path to goal (through Rimnicu) since it is only focused on the cost remaining rather than the total cost.

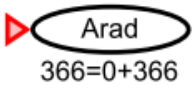
# GREEDY BEST-FIRST PROPERTIES

- **Not complete -** may follow an infinite path.
  - Most reasonable heuristics will not cause this problem however.
- Worst case **time complexity** is still $\mathbf{O(b^m)}$ where $m$ is the maximum depth.
  - A good heuristic will avoid the worst-case behavior for most problems.
- **Space-complexity** is also $\mathbf{O(b^m)}$ - must maintain a queue of all unexpanded states

# A* SEARCH

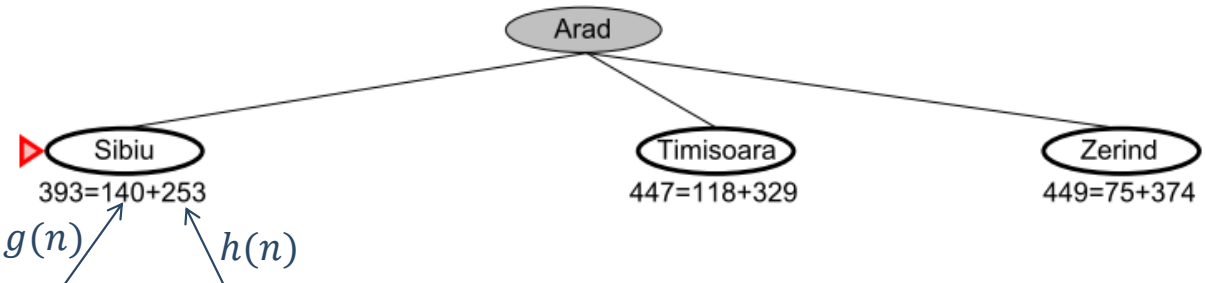
- A* selects which node to expand next by using an evaluation function $f$.
- This function is a combination of the cost of getting to the node and the heuristic evaluation of the node: $f(n) = g(n) + h(n)$
- The value $f(n)$ is the *estimated* cost of the cheapest solution that goes through node $n$.
- Note: A* works for any search problem, but the heuristic function $h$ is problem specific.

# A* SEARCH EXAMPLE




| Straight−line distance to Bucharest | |
|---|---|
| **Arad** | 366 |
| **Bucharest** | 0 |
| **Craiova** | 160 |

# A* SEARCH EXAMPLE

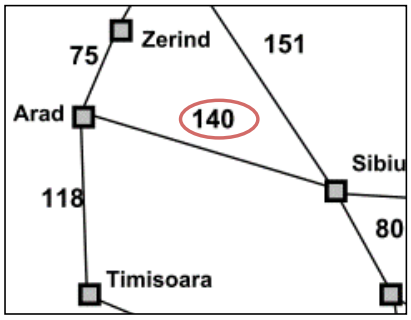



$g(n)$

$h(n)$

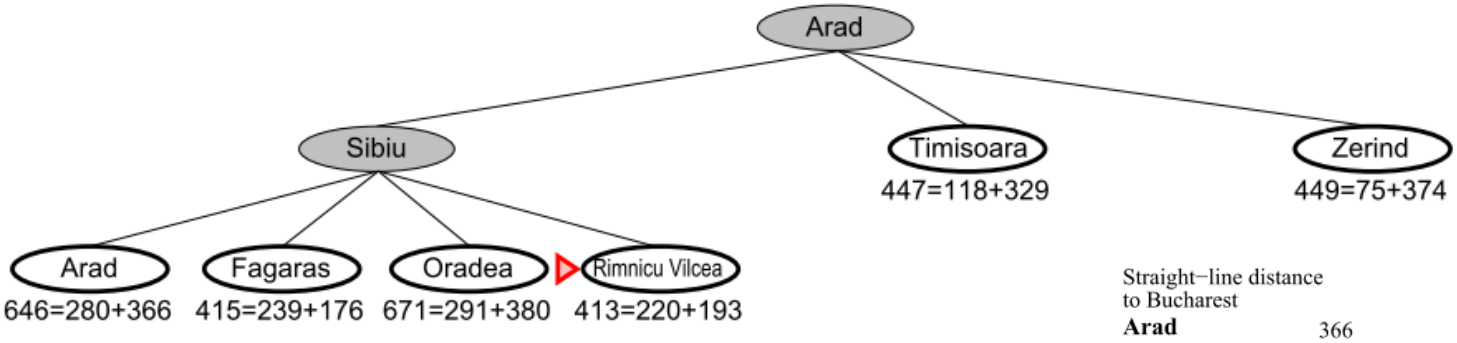
**Actual distance Arad - Sibu**

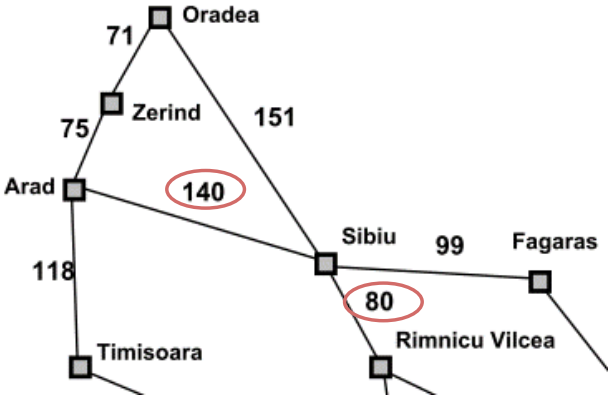
**Straight line distance Sibu - Bucharest**




| Straight−line distance to Bucharest | |
|---|---|
| **Sibiu** | 253 |

# A* SEARCH EXAMPLE

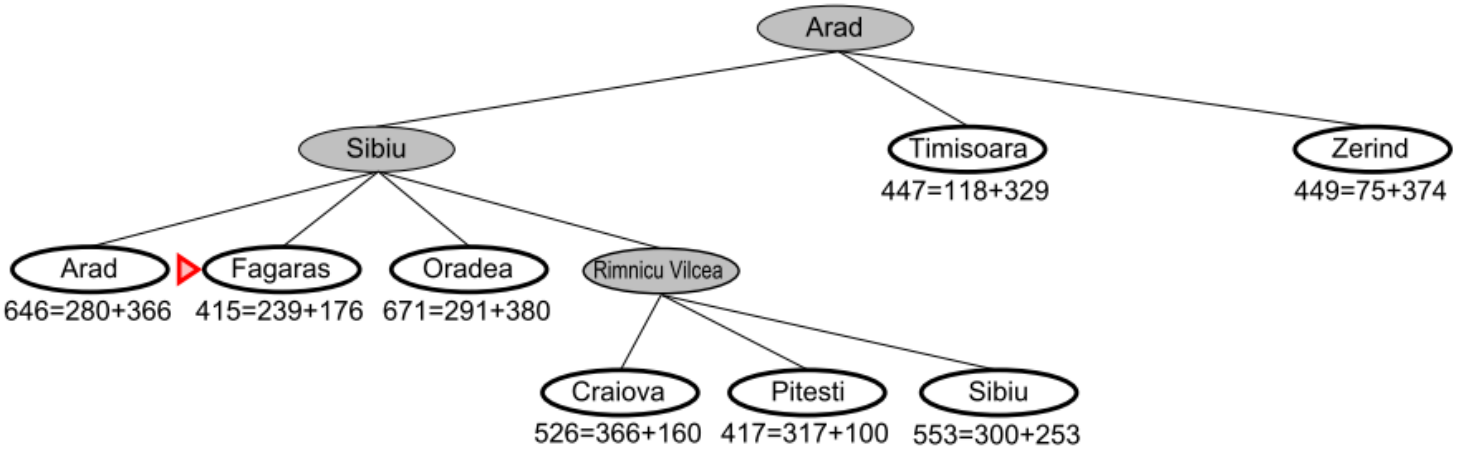


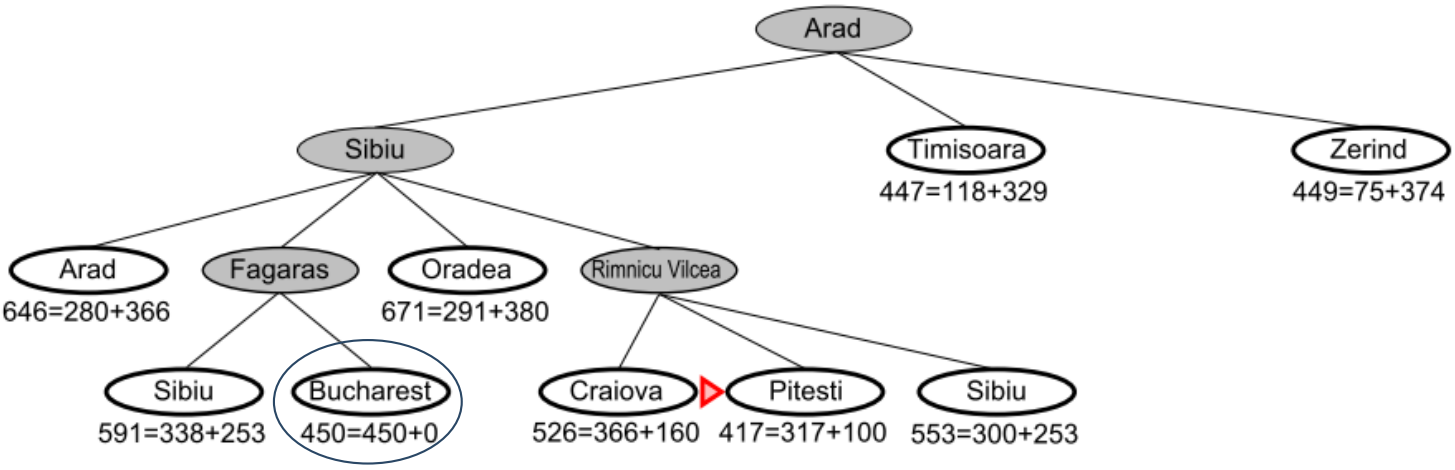

Straight−line distance to Bucharest

| | |
|---|---|
| **Arad** | 366 |
| **Bucharest** | 0 |
| **Craiova** | 160 |
| **Dobreta** | 242 |
| **Eforie** | 161 |
| **Fagaras** | 178 |
| **Giurgiu** | 77 |
| **Hirsova** | 151 |
| **Iasi** | 226 |
| **Lugoj** | 244 |
| **Mehadia** | 241 |
| **Neamt** | 234 |
| **Oradea** | 380 |
| **Pitesti** | 98 |
| **Rimnicu Vilcea** | 193 |
| **Sibiu** | 253 |
| **Timisoara** | 329 |
| **Urziceni** | 80 |
| **Vaslui** | 199 |
| **Zerind** | 374 |

# A* SEARCH EXAMPLE

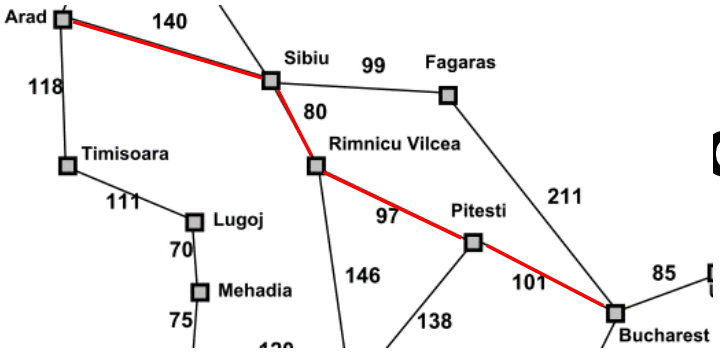


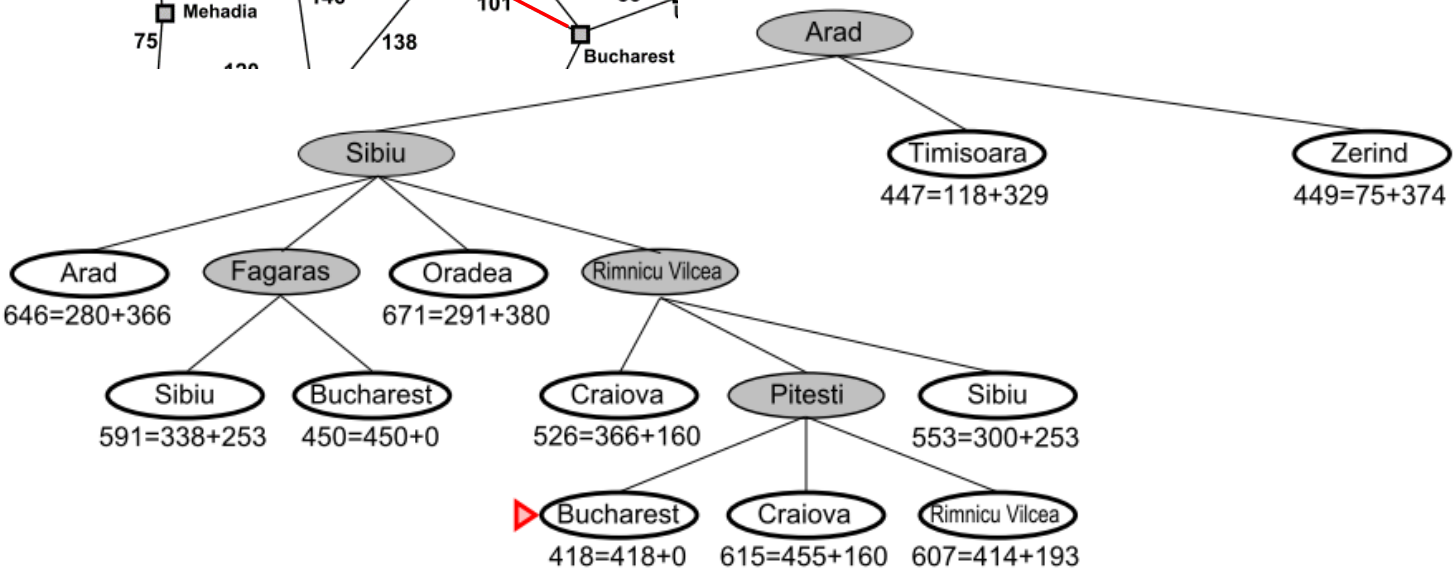
# A* SEARCH EXAMPLE




**Note that we reached Bucharest, but another path seems to be shorter, so we must check it out!**

# CH EXAMPLE




**No path can be shorter than this!**

# A* OPTIMALITY

- A* tree search is optimal with any admissible heuristic.
  - Admissible = **never overestimates the cost**
- A* graph search is optimal with any consistent heuristic.
  - Consistent = stronger requirement than admissible (≈ taking a detour cannot be cheaper)
  - Most admissible heuristics are also consistent
- A* is optimally efficient for any consistent heuristic function $h$.
  - I.e. no other optimal strategy expands fewer nodes than A* when using the same $h$.
- A major drawback with A* (and other graph-based search algs) is that it runs out of memory for large problems
  - Generally runs out of memory before running out of time.

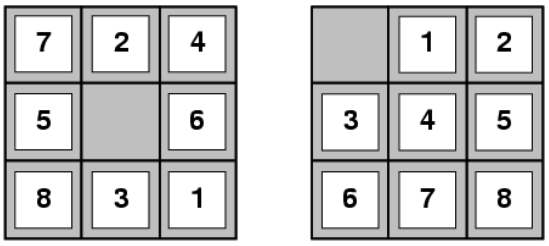
# A* SPEED VS. ACCURACY

- If $h(n) = 0$ for all nodes $n$, A* is the same as Uniform Cost Search. It will always find an optimal path, but it may take a while.
  - Accurate
- If $h(n)$ is the exact cost of the cheapest path from $n$ to a goal state, A* only searches the optimal path.
  - Fast and accurate (but unrealistic)
- If $h(n)$ overestimates the cost of getting to a goal state, A* will be fast, but will not always find the shortest path.
  - Fast, but **not** accurate
- We see that the performance of a heuristic search alg. depends on the quality of $h(n)$

# A* PERFORMANCE

- Typical search costs (average number of nodes expanded) for 8-puzzle:
  - $h_1$: Number of tiles out of place
  - $h_2$: Manhattan distance (i.e., no. of squares from the desired location of each tile)

| Depth | IDS | $A^*(h_1)$ | $A^*(h_2)$ |
|---|---|---|---|
| 2 | 10 | 6 | 6 |
| 12 | 3 644 035 | 227 | 73 |
| 24 | - | 39 135 | 1 641 |




Start State Goal State

# SUMMARY SEARCH ALGORITHMS

- Implemented by maintaining a queue of unexpanded search nodes.
- Can be informed or uniformed
  - Informed use heuristics. Most known: A*
- Judged based on **completeness**, **optimality**, **time complexity**, and **space complexity**.
  - Complexity depends on branching factor *b* and the depth *d*
- Performance is a trade-off between path cost and search cost
- Performance of heuristic search algorithms depends on the quality of the heuristic

# COMPARISON TABLE

| | BFS | DFS | IDS | A* |
|---|---|---|---|---|
| **Complete?** | Yes | No | Yes | Yes |
| **Optimal?** | Yes[1] | No | Yes[1] | Yes[2] |
| **Time** | O(bd) | $O(b^d)$ | $O(b^d)$ | $O(b^d)$ |
| **Space** | $O(b^d)$ | $O(b^m)$ | $O(b^m)$ | $O(b^d)$ |

[1]if all actions have same cost
[2]with admissible heuristic.

Note! Time complexity of A* depends on the heuristics

# CLASSICAL PLANNING

## R&N CHAPTER 10

**What is planning?**

**How to represent plans**

**Searching for plans**

UMEÅ UNIVERSITY

# PLANNING

- **Planning** - devising a plan of action to achieve one's goals - is a central part of AI
  - Used for large logistics problems, operational planning, robotics, scheduling etc.
  - Bi-annual planning competition
  - Several international Conferences on Planning
- So far, we have looked at two ways to do planning:
  - search-based problem-solving agent (Ch. 3) – needs domain-depended heuristics
  - hybrid logical agent (Ch. 7) – suffer from combinatorial explosion for bigger problems
- Now we will combine the two to handle larger problems

# CLASSICAL PLANNING

- In FOL we need to specify what changes *and* what stays the same as the result of an action -> the **frame problem**
- Classical planning focuses on problems where most things stays the same
  - We need another representation
  - Solution: **PDDL** - Planning Domain Definition Language, a subset of FOL

# PDDL

- PDDL is derived from the STRIPS planning language.
  - More on STRIPS in upcoming lecture
- Contains:
  - Initial state and goal state.
  - A set of *Actions*(*s*) in terms of preconditions and effects.
  - **Closed world assumption**: Unmentioned state variables are assumed false.

$$
\begin{aligned}
&Action(Fly(P_1, SFO, JFK),\\
&\quad \textsc{Precond}: At(P_1, SFO) \wedge Plane(P_1) \wedge Airport(SFO) \wedge Airport(JFK)\\
&\quad \textsc{Effect}: \neg At(P_1, SFO) \wedge At(P_1, JFK))
\end{aligned}
$$

# EXAMPLE: AIR CARGO TRANSPORT

- A classical transportation problem: Loading and unloading cargo and flying between different airports.

**Actions**: Load(cargo, plane, airport), Unload(cargo, plane, airport), Fly(plane, airport, airport)

**Predicates**: In(cargo, plane), At(cargo ∨ plane, airport)

<u>Example solution:</u>

Load(C1, P1, SFO), Fly(P1, SFO, JFK), Unload(C1, P1, JFK),
Load(C2, P2, JFK), Fly(P2, JFK, SFO), Unload(C2, P2, SFO).

# PDDL

- Intended to be a standard for describing a certain *planning domain* (not a specific plan)
- Restricted language -> efficient algorithm
- Several implementations of PDDL exists
- Possible to construct software for very advanced plans, with many preconditions and effects for different actions
  - The designers of the PDDL solver does not need to know the domain in which the software is going to be used
  - Companies that need planning software can define their planning domains in PDDL, without aligning too hard to a specific implementation

# STATE SPACE SEARCH

- Planning can be done by searching the sate space (defined by PDDL for example)
- Forward (progression):
  - state-space search considers actions that are *applicable* (could be *next* part of the plan leading up to the current goal state)
- Backward (regression):
  - state-space search considers actions that are *relevant* (could be the *last* part of the plan)
- Neither of them is efficient without good heuristics!
  - Can be derived automatically

# SUMMARY PLANNING

- PDDL is a language for representing (simple) planning problems
  - Builds on FOL but more restricted
- State space search (forward or backward) can be used to search for a plan
- Real-world planning and scheduling (e.g. operations of spacecrafts, factories, and military campaigns) is more complex
  - Both the representation language and the way the planner interacts with the environment needs to be extended
  - Beyond the scope of this course

# Setting up

- **No zoom chat**
  - Questions will be answered at specific times
- **Suppress distractions**
  - Clear notifications
    - Turn off your phone, mails, Facebook…
- Get ready: Open a clean browser with only:
  - Your personal report
  - The course instructions: https://tinyurl.com/instructions-fund-of-ai

# ARTIFICIAL NEURAL NETWORKS

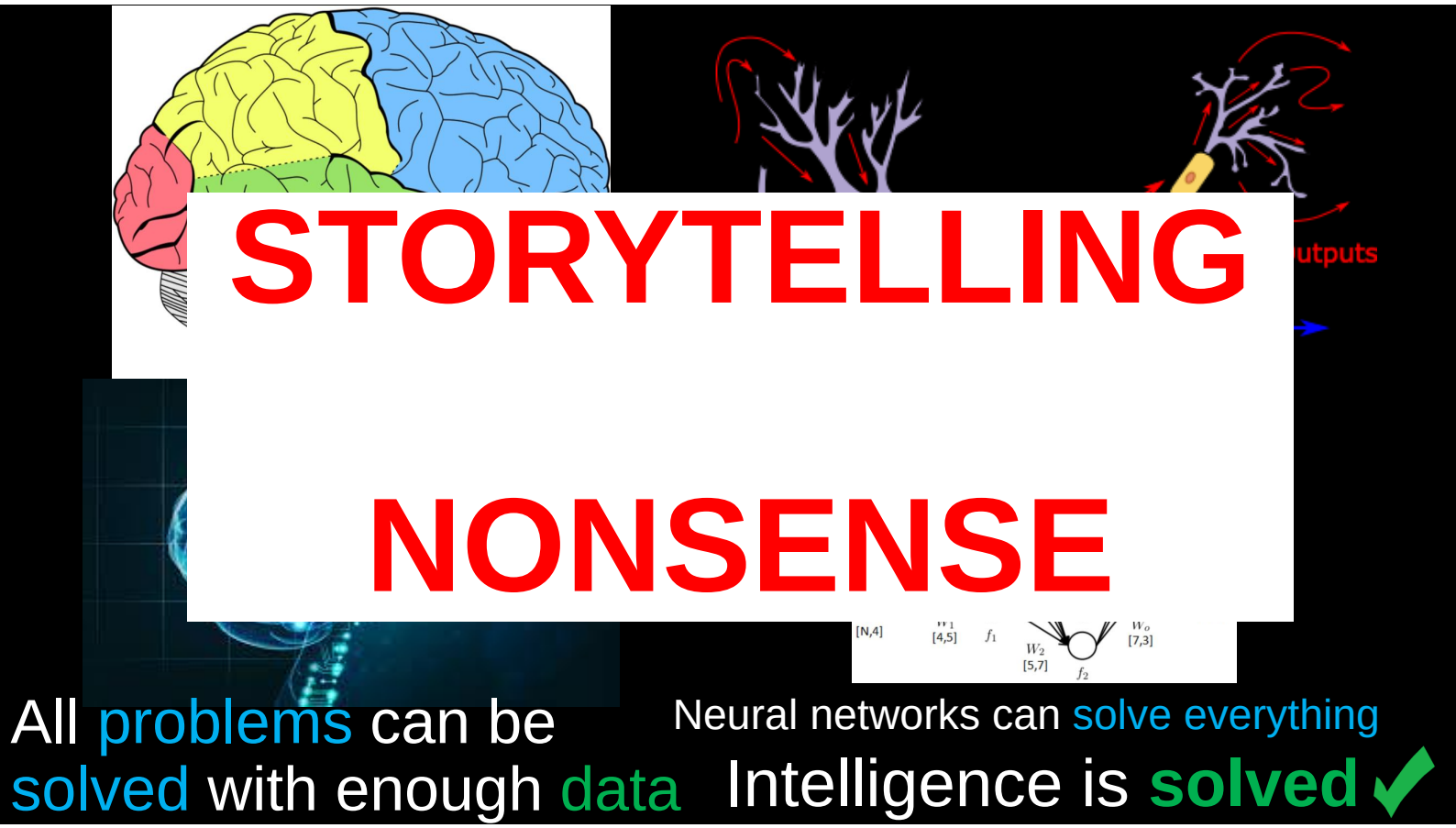
Loïs Vanhée
Associate professor
Responsible and Ethical Artificial Intelligence
loisv@cs.umu.se

5DV124,5DV201
Fundamentals of Artificial Intelligence

Department of Computing Science

UMEÅ UNIVERSITY

# What is generally said about neural networks?



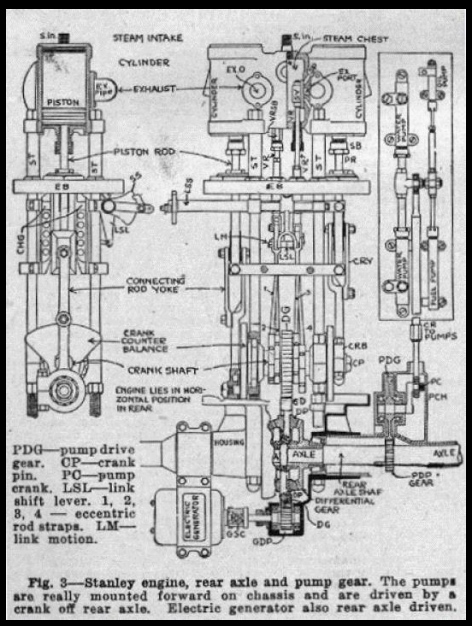

All problems can be solved with enough data

Neural networks can solve everything

Intelligence is **solved** ✓

# Tool thinking

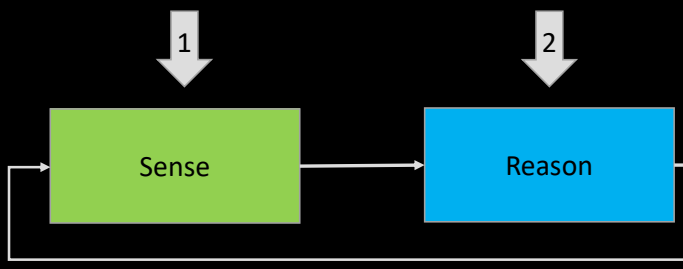
Tool in the toolbox

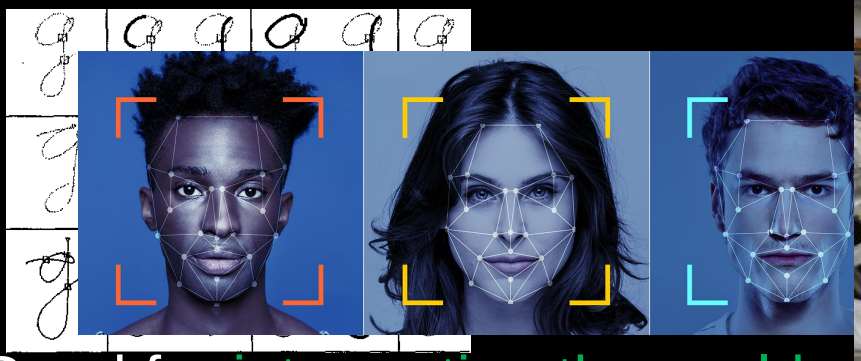



Inners of the tool

Usage and costs

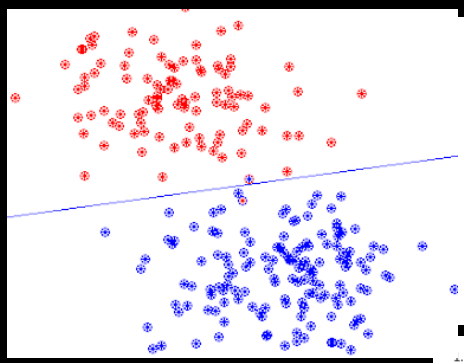
# Neural networks within the int

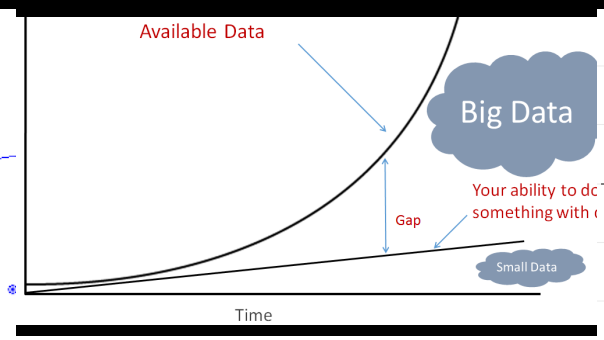


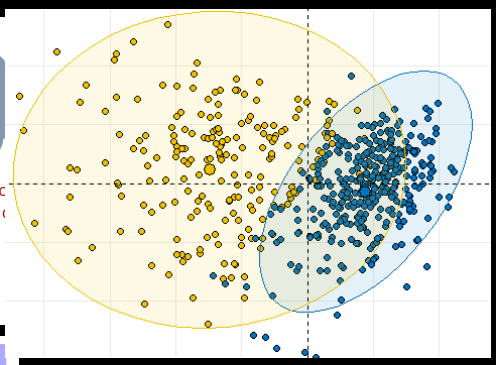


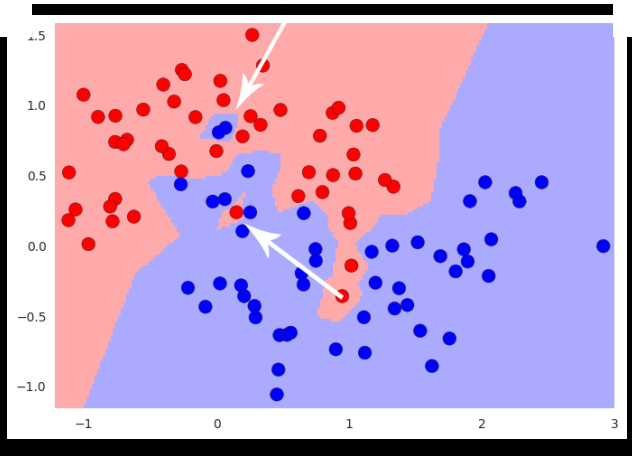
Good for interpreting the world
Bad for elaborated thinking

# In the machine learning toolbox

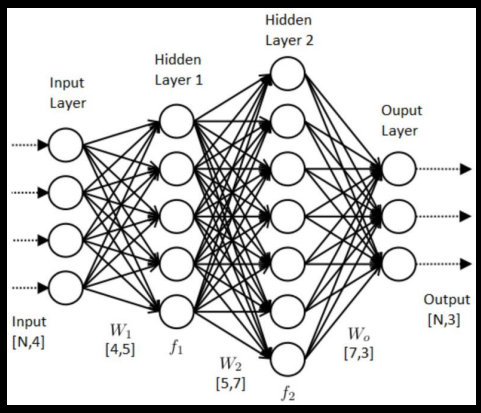


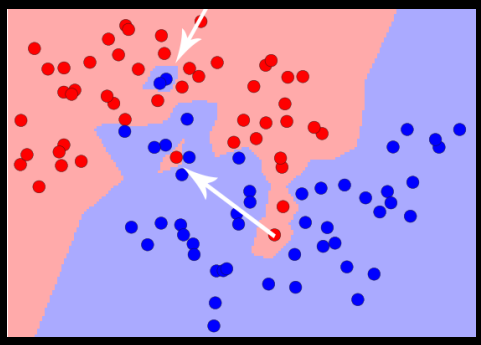


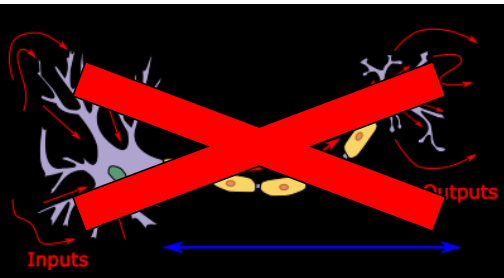





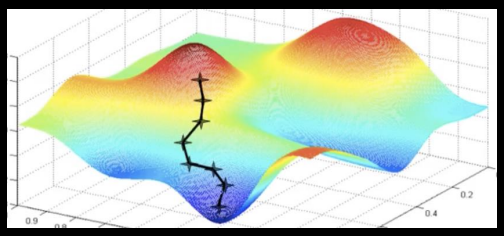
# Inner workings



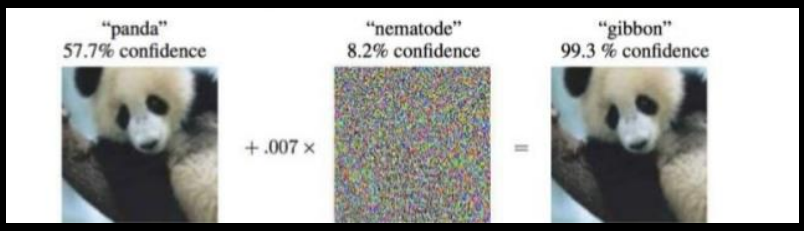


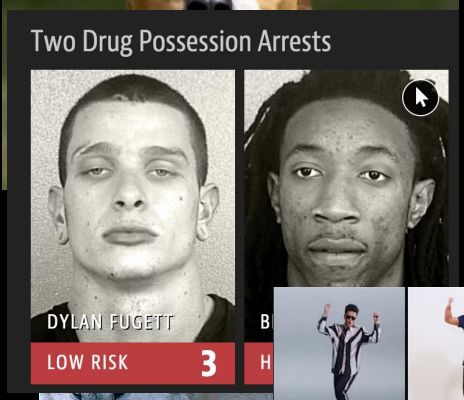

Search
Gradient descent



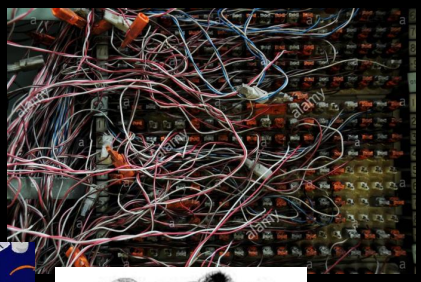
For math, algorithms details, re-check the references

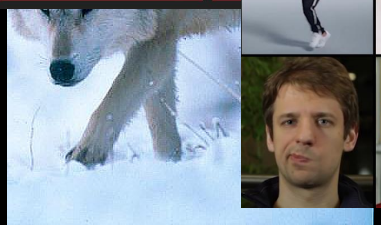
# Costs and issues

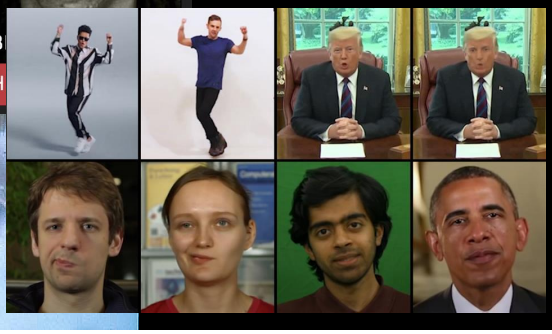


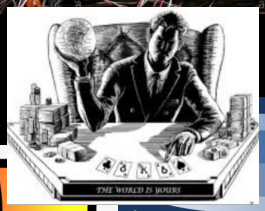


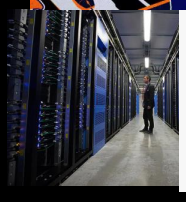

Sensitive to bias
Fragile and hard to fix
Very expensive
Raises serious social issues



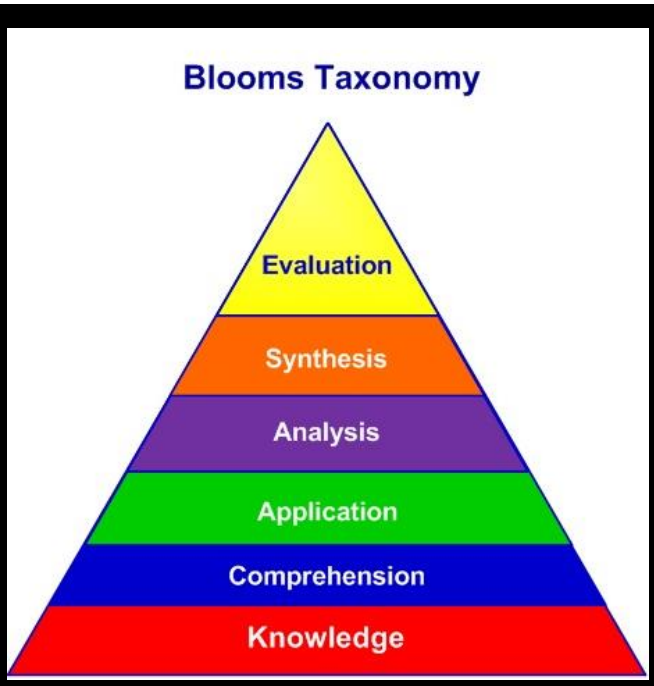








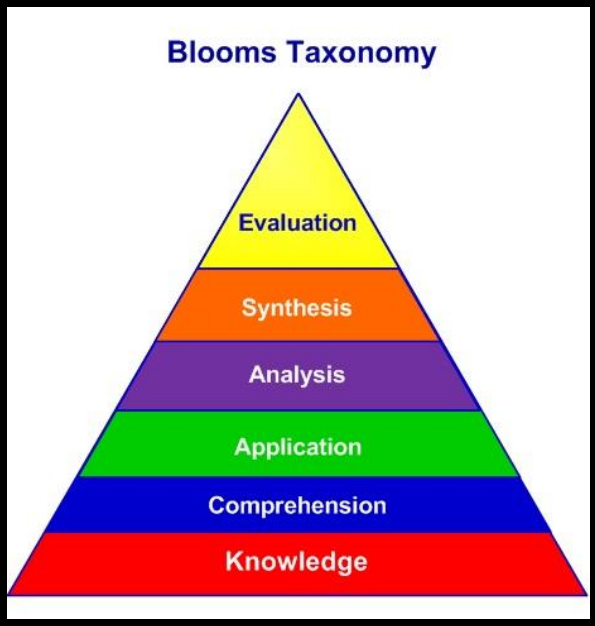
Your turn to play!




Real-world skill tree

# Intended learning outcomes

At the end of the session, you should be able to:




**Define** the key concepts of neural networks

**Relate** these key concepts with each other and other AI tools

**Manipulate** neural network on a pre-set concrete example

**Sketch** how to implement neural networks for a specific problem

**Appraise** how neural Networks can be applied in a given problem

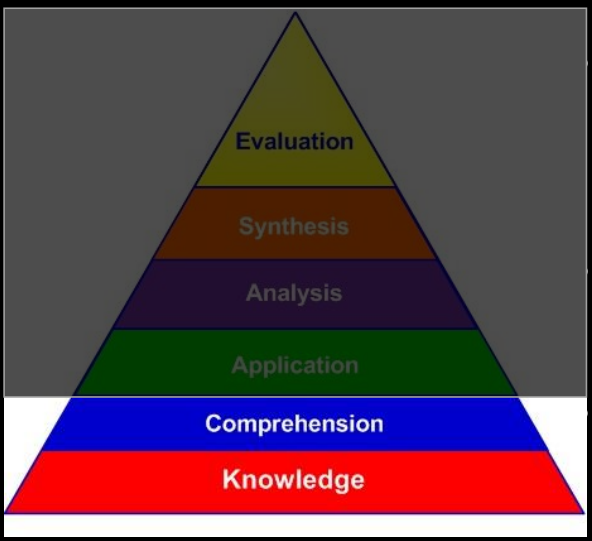
## Same as last time: discussions in small groups

- Breakout rooms for ~3x15 minutes
  - Discuss your answers for each part
    - Keep the conversation flowing
  - **10 minutes:** break and questions
- Check the instructions, if it is not already open:

  https://tinyurl.com/instructions-fund-of-ai

No chitchat, be effective

# Part A




- **Define** the key concepts of neural networks
- **Relate** these key concepts with each other and other AI tools
- **Manipulate** neural network on a pre-set concrete example
- **Sketch** how to implement neural networks for a specific problem
- **Appraise** how neural Networks can be applied in a given problem

**https://tinyurl.com/fundOfAI**

Turn on your micro and camera when your question is picked up

Please write your name

# On an ideal white board (and in your mind in the exam)

If you cannot come up with, define and relate these concepts, consolidate them during the post-class

Neuron
Weight
Bias
Input/output layers
Hidden layers
Sigmoid function
Backpropagation
Convolutional NN
Deep NN
DFF, RNN, LSTM

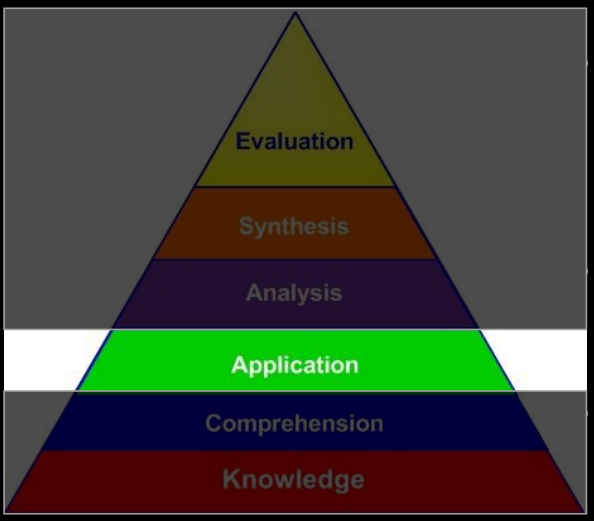
Databases/ datafarms
High-performance computing

Pattern recognition
Test loss
Training loss
Features
Training data (cross-validation)
Overtraining/overfitting
Black box

Classification
Gradient descent
Grounding problem

Data
Data collection
Computation time
Energy expenses
Hardware expenses

# Second activity: Application

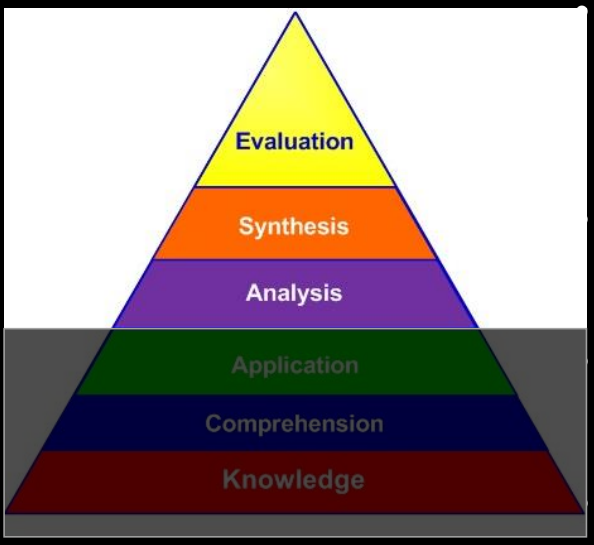



- **Define** the key concepts of neural networks
- **Relate** these key concepts with each other and other AI tools
- **Manipulate** neural network on a pre-set concrete example
- **Sketch** how to implement neural networks for a specific problem
- **Appraise** how neural Networks can be applied in a given problem

**https://tinyurl.com/fundOfAI**

Turn on your micro and camera when your question is picked up

# Third activity: Analysis




**Define** the key concepts of neural networks

**Relate** these key concepts with each other and other AI tools

**Manipulate** neural network on a pre-set concrete example

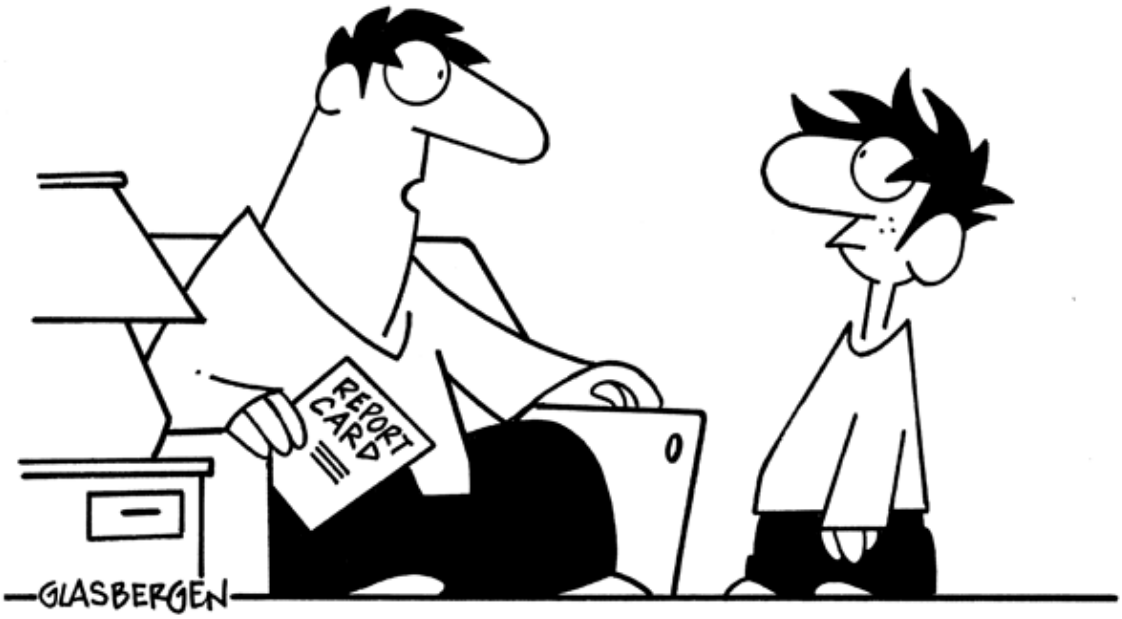
**Sketch** how to implement neural networks for a specific problem

**Appraise** how neural Networks can be applied in a given problem

**https://tinyurl.com/fundOfAI**

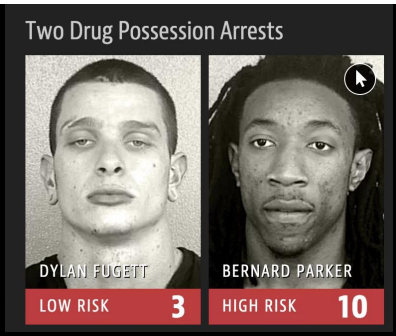
Turn on your micro and camera when your question is picked up

© Randy Glasbergen
www.glasbergen.com




**"I probably remember 20% of the stuff I learned in school and forgot the other 90%."**

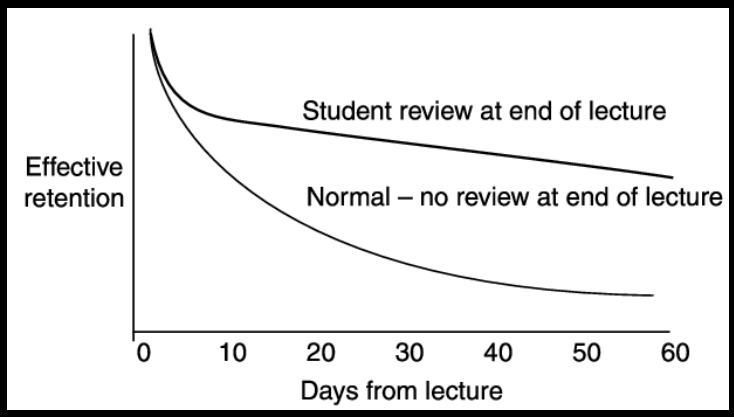
# Take home message




- Neural networks are vaguely inspired from human neurons but they are widely different. They just use data for greedily adjusting many parameters
- Neural networks excel for input recognition
- Neural network hype is a fallacy: they just fixed the last problem of a long chain
- Neural networks are fragile: unexplainable arbitrary answers
- Neural networks are a sensitive tool: financial costs (power games), prone to bias, they generate a lot of social inequality

# Anchoring

https://tinyurl.com/fundOfAI-LR




This is anchoring:
State what you learned

Write your answers on a sheet of paper if you can (though it helps us to know what you learned)

Save your feedback for later

# Setting up

- **No zoom chat**
  - Questions will be answered at specific times
- **Suppress distractions**
  - Clear notifications
    - Turn off your phone, mails, Facebook…
- Get ready: Open a clean browser with only:
  - Your personal report
  - Course instructions: https://tinyurl.com/instructions-fund-of-ai

# EMBODIED AI
# ARTIFICIAL EVOLUTION
# EMERGENCE

Loïs Vanhée
Associate professor
Responsible and Ethical Artificial Intelligence
loisv@cs.umu.se

5DV124,5DV201
Fundamentals of Artificial Intelligence

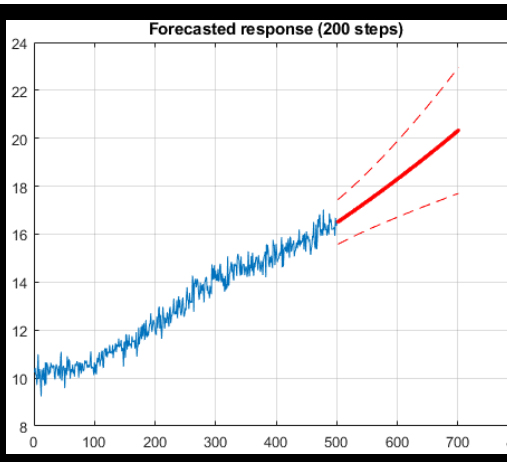
Department of Computing Science

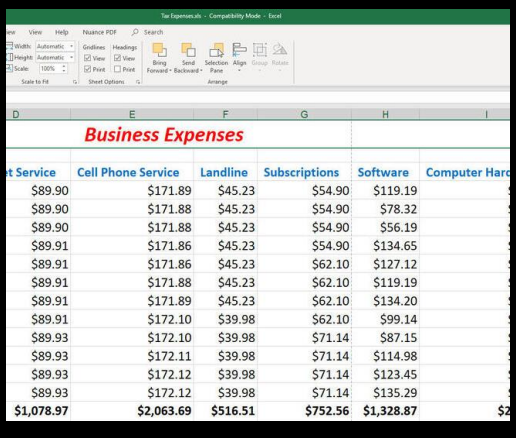
UMEÅ UNIVERSITY

# Cognitive tasks can be more or less easy or hard

# Is that true?

# Some complex tasks for humans can be easy for computers



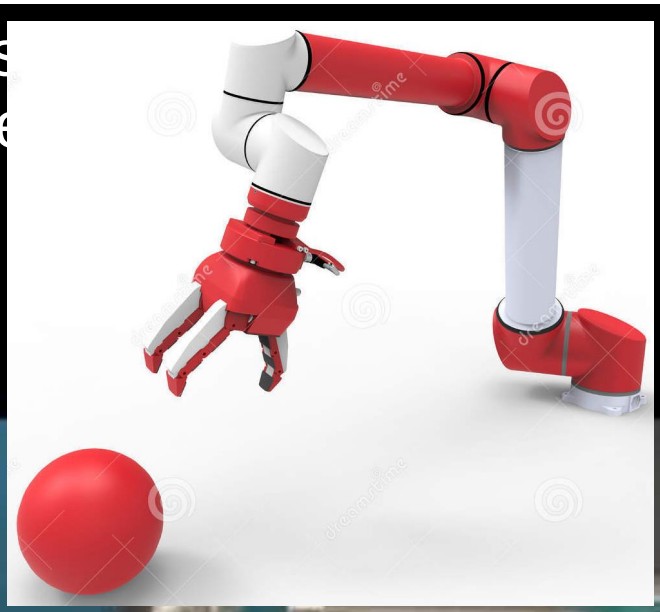
Business Expenses

| t Service | Cell Phone Service | Landline | Subscriptions | Software | Computer Harc |
|---|---|---|---|---|---|
| $89.90 | $171.89 | $45.23 | $54.90 | $119.19 | $ |
| $89.90 | $171.88 | $45.23 | $54.90 | $78.32 | $ |
| $89.90 | $171.88 | $45.23 | $54.90 | $56.19 | $ |
| $89.91 | $171.86 | $45.23 | $54.90 | $134.65 | $ |
| $89.91 | $171.86 | $45.23 | $62.10 | $127.12 | $ |
| $89.91 | $171.88 | $45.23 | $62.10 | $119.19 | $ |
| $89.91 | $171.89 | $45.23 | $62.10 | $134.20 | $ |
| $89.91 | $172.10 | $39.98 | $62.10 | $99.14 | $ |
| $89.93 | $172.10 | $39.98 | $71.14 | $87.15 | $ |
| $89.93 | $172.11 | $39.98 | $71.14 | $114.98 | $ |
| $89.93 | $172.12 | $39.98 | $71.14 | $123.45 | $ |
| $89.93 | $172.12 | $39.98 | $71.14 | $135.29 | $ |
| $1,078.97 | $2,063.69 | $516.51 | $752.56 | $1,328.87 | $2 |

Some easy tasks for humans
can be very hard for compute

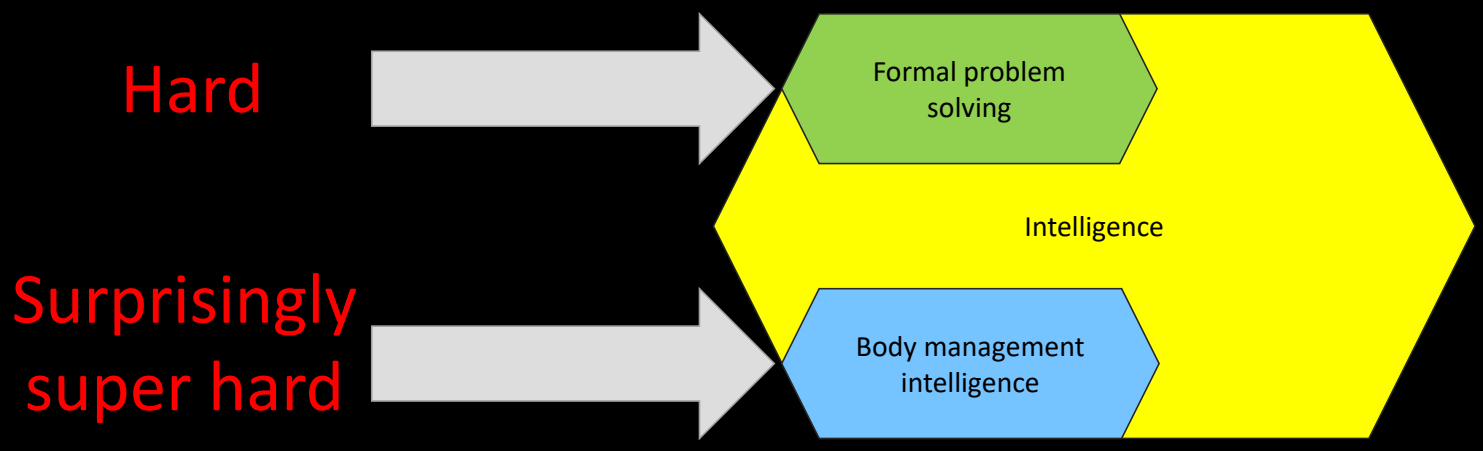
```
climb(stairs)
{
        while(any_more_step_in_stairs())
        {

}
```

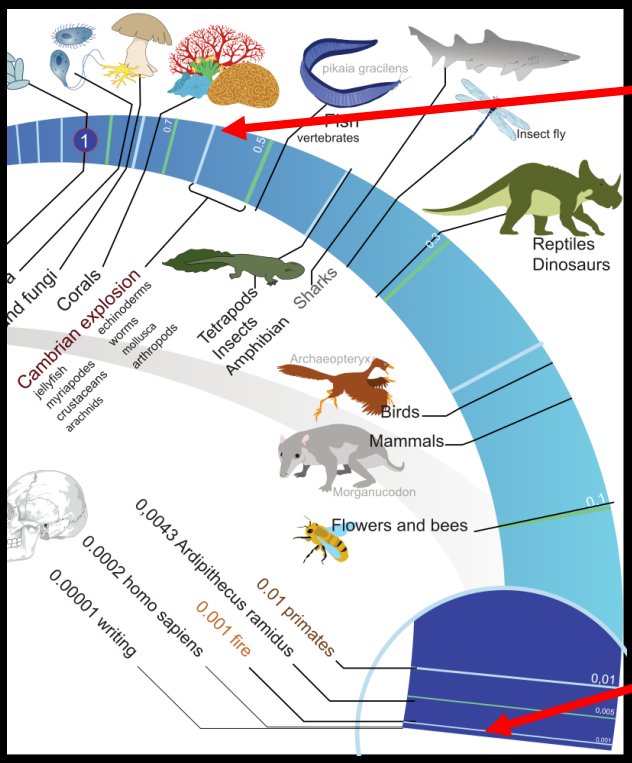


https://www.youtube.com/watch?v=VTlV0Y5yAww

# Millennia conceiving problems difficulty… Just got broken by AI?



# Maybe not!




Time since body management is important

And maybe robot body (motors, hard pieces) is less fit for the task

Time since mathematics are important

**Human and artificial intelligences are inherently shattered**

**Just not the same way**

**Can you think in 4D? 5D?**
**Can you open a door?**

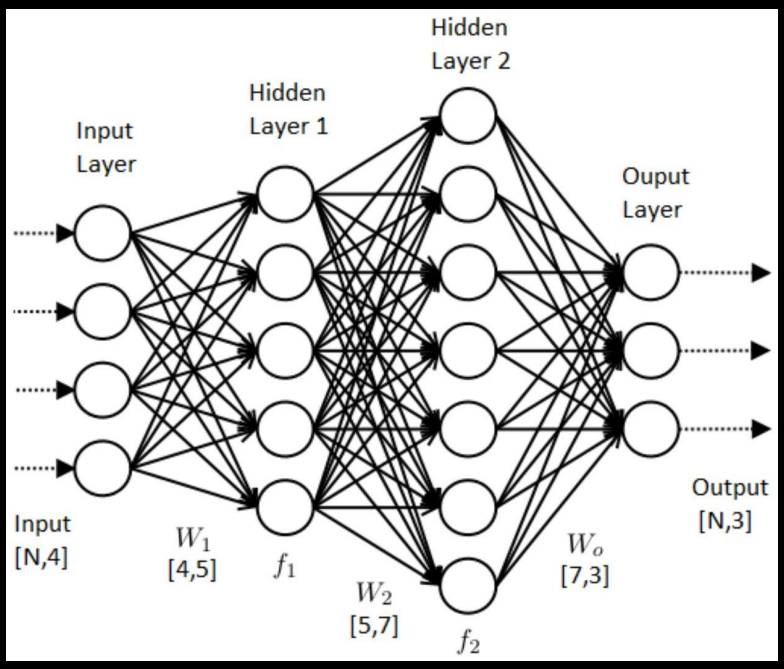
uter un pied de page

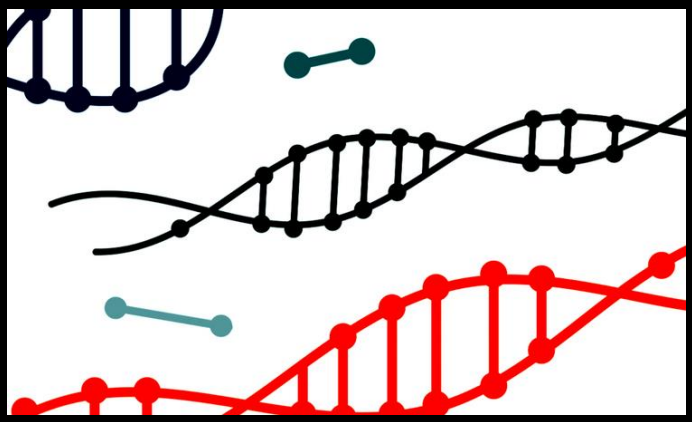
AI is very bad at « body intelligence »

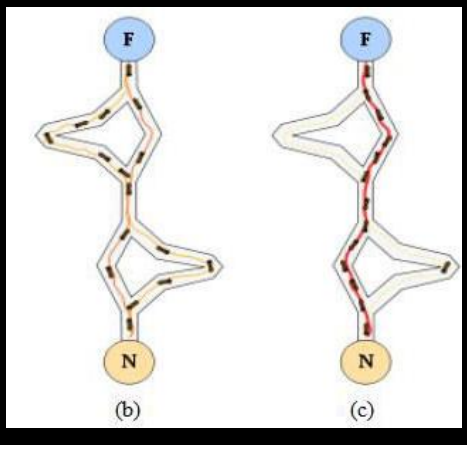
**Inspiration for building well-working embodied systems?**

Artificial life
Evolution

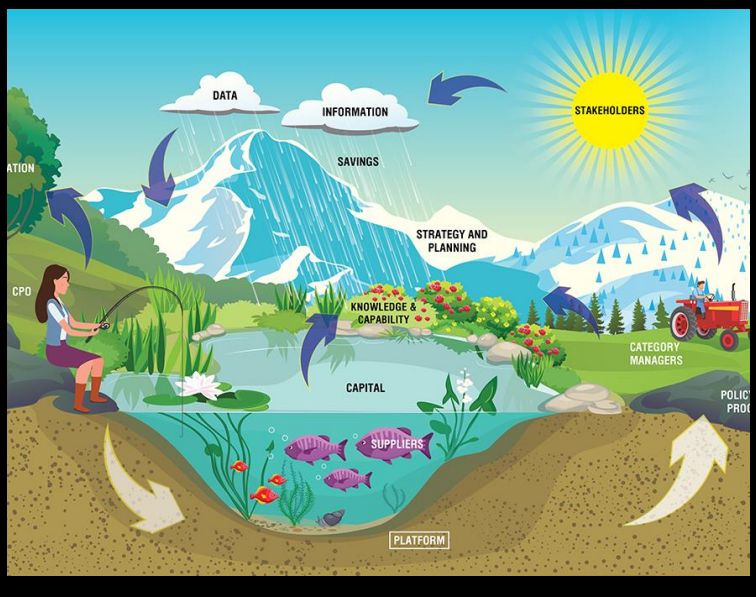
**Nature makes workable embodied intelligent systems**

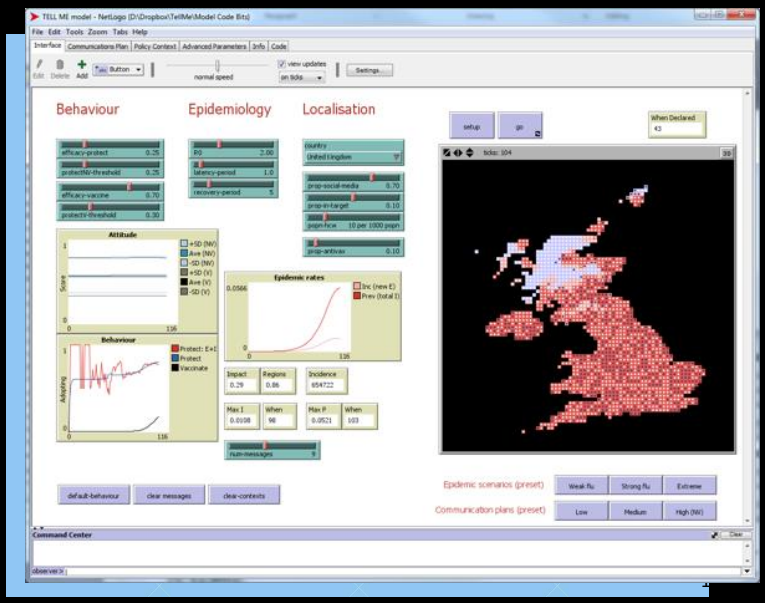

Hidden
Layer 2
Hidden
Layer 1
Input
Layer
Ouput
Layer
Output
[N,3]
Input
[N,4]
$W_1$
[4,5]
$f_1$
$W_2$
[5,7]
$f_2$
$W_o$
[7,3]




F
N
(b)
F
N
(c)

# In return, replicating with AI can help understanding nature & society



# We have so much to learn from each other!

**AI inspired by humans/nature**

- Planning
- Search
- Learning
- Neural nets
- …

**Still pretty hard to replicate for AI**

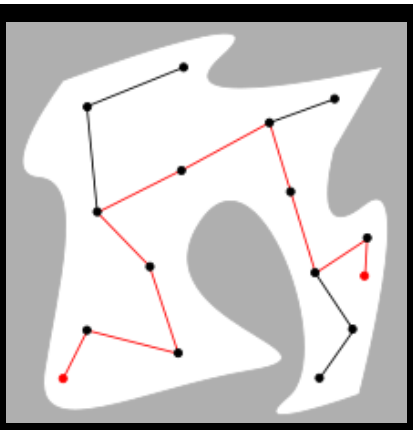
- Motion/body intelligence
- General intelligence
- Common sense

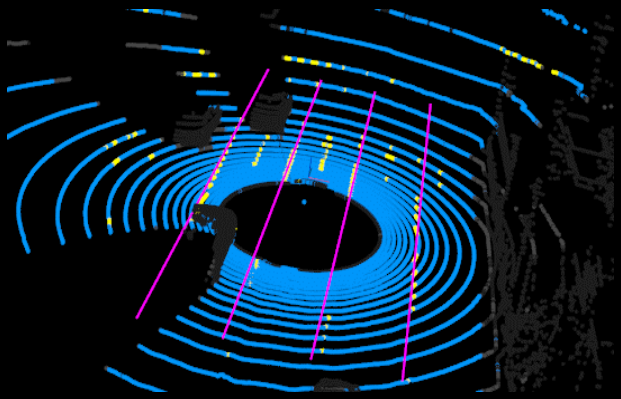
**Using AI for understanding nature**

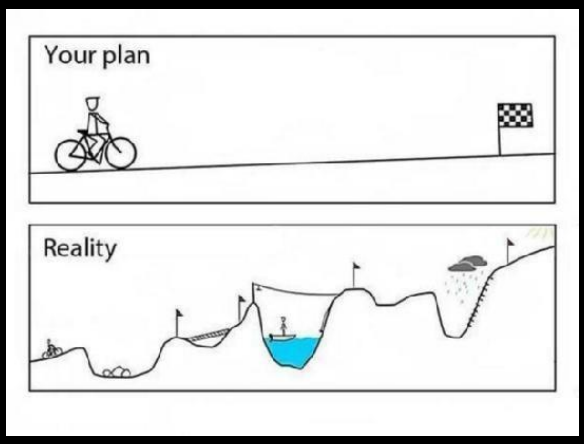
- Simulation
- Modelling

**What is at play in embodied intelligence**

Hint: more than body and intelligence!

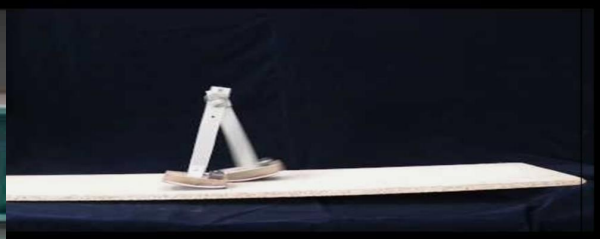





Your plan
Reality

J'STAR

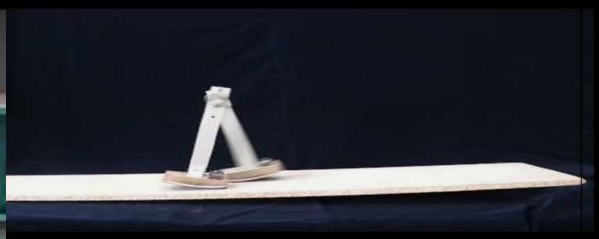
# The right body can lower cognitive needs (or cognitive challenge)



https://www.youtube.com/watch?v=4PaTWufUqqU

https://www.youtube.com/watch?v=CK8IFEGmiKY

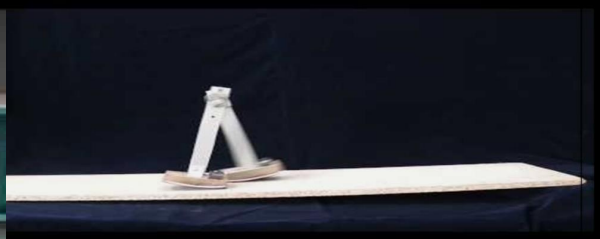
The right body in the right environment can lower cognitive needs (or cognitive challenge)



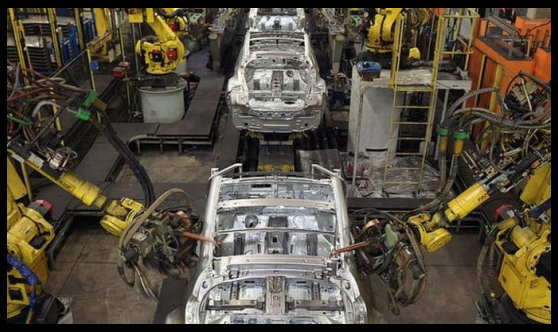
https://www.youtube.com/watch?v=4PaTWufUqqU

https://www.youtube.com/watch?v=CK8IFEGmiKY

The right body in the right environment and right socioecological niche can lower cognitive needs (or cognitive challenge)



https://www.youtube.com/watch?v=4PaTWufUqqU



https://www.youtube.com/watch?v=CK8IFEGmiKY

The right body in the right environment and right socioecological niche can lower cognitive needs (or cognitive challenge)

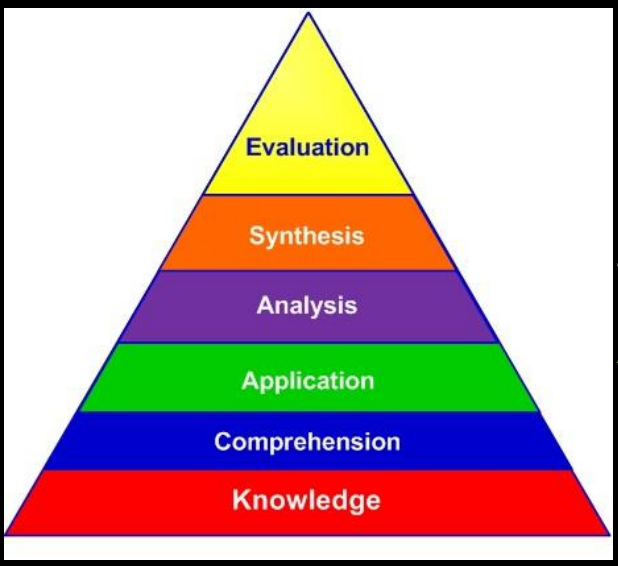
- An engineering treasure
- But
  - In the wrong environment
  - With a body unfit for the task
  - Trying to resolve the problem with insufficient support

- **Today we are going to learn how to avoid such traps**
- **And what bio-inspired systems can bring and offer**

# Embodied AI
# Artificial Evolution
# Emergence

# Your turn to play

# Intended learning outcomes



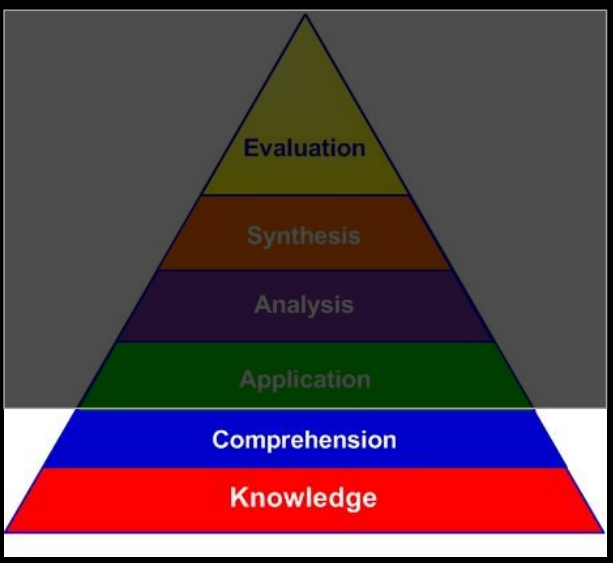

List and define the key concepts related to **artificial life, emergence, and embodied AI**

Relate these concepts with each other and those of **other courses & disciplines**

Apply **Genetic Algorithms** for solving concrete problems

Create a sketch of an Artificial Life

# Intended learning outcomes




List and define the key concepts related to **artificial life, emergence, and embodied AI**

Relate these concepts with each other and those of **other courses & disciplines**

Apply **Genetic Algorithms** for solving concrete problems

Create a sketch of an Artificial Life

**https://tinyurl.com/fundOfAI**

Turn on your micro and camera when your question is picked up

Please write your name

# On an ideal white board (and in your mind in the exam)

If you cannot come up with, define and relate these concepts, consolidate them during the post-class

Artificial Life
Mind
Body
Environment
Ecosystem
Niche
Evolution

Adaptability
Robustness

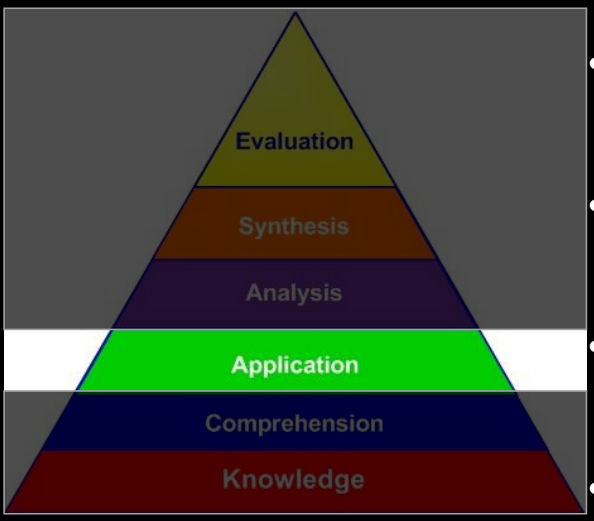
Social simulation
Social sciences
Cognitive sciences

Mind-body-environment-niche relation

Emergence
Stygmergy
Neural networks
Symbol grounding problem

Embodied agent
Sensor
Effector
Physics

Generic algorithm
Genotype
Phenotype
Selection
Mutation
Reproduction

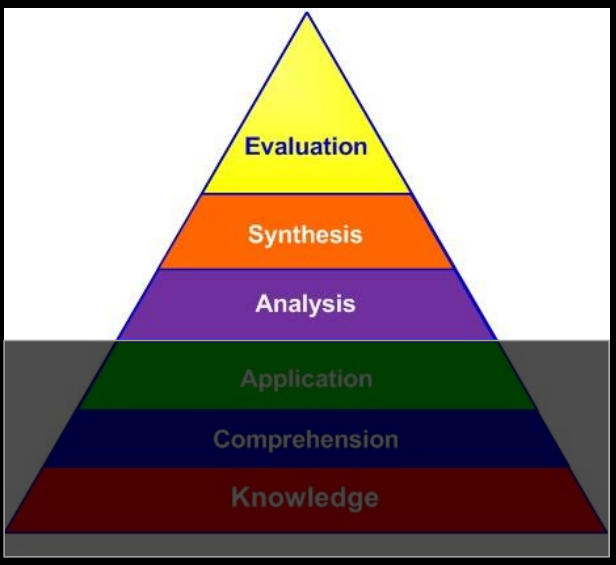



- List and define the key concepts related to **artificial life, emergence, and embodied AI**
- Relate these concepts with each other and those of **other courses & disciplines**
- Apply **Genetic Algorithms** for solving concrete problems
- Create a sketch of an Artificial Life

**https://tinyurl.com/fundOfAI**

Turn on your micro and camera when your question is picked up

Please write your name




List and define the key concepts related to **artificial life, emergence, and embodied AI**

Relate these concepts with each other and those of **other courses & disciplines**

Apply **Genetic Algorithms** for solving concrete problems

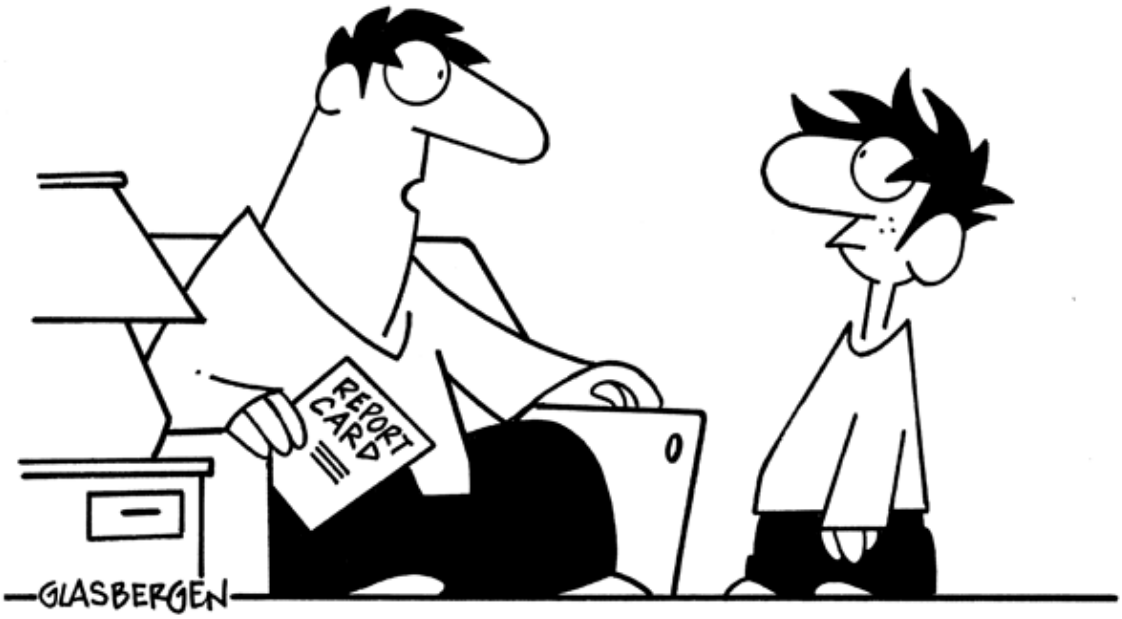
Create a sketch of an Artificial Life

**https://tinyurl.com/fundOfAI**

Turn on your micro and camera when your question is picked up

Please write your name

© Randy Glasbergen
www.glasbergen.com




**"I probably remember 20% of the stuff I learned in school and forgot the other 90%."**

## Take home message

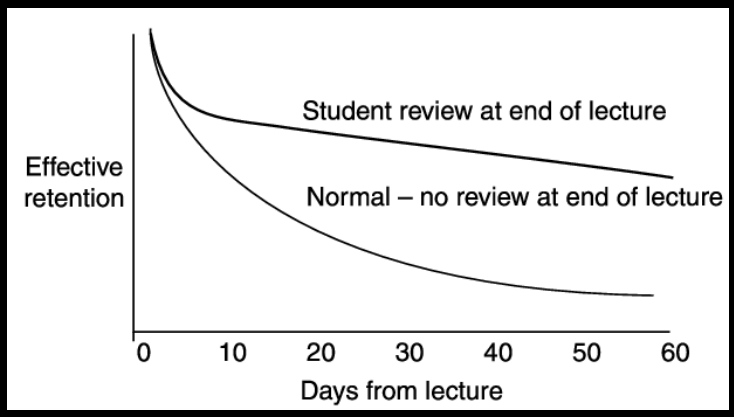
- Body management is a hard problem
- Bio/socio-inspired systems is a great approach for solving AI problems
- Artificial life is a great application of AI for other disciplines
- Body, mind, eco/sociosystems, and environments are deeply interleaved
- Emergence involves a new level of interpretation that changes system dynamics. Emergence is frequent in AI, from collective intelligence to consciousness

# If you want to learn more

- Checkout online resources on Canvas
- Mingle with your colleagues
  - Go talk to your colleagues
  - Checkout their report
  - Checkout which artificial life they created ☺
- Implement an agent society
- Teach agents to walk using genetic algorithms

# Anchoring

https://tinyurl.com/fundOfAI-LR




This is anchoring:
State what you learned

Write your answers on a sheet of paper if you can (though it helps us to know what you learned)

Save your feedback for later

# Setting up

- **No zoom chat**
  - Questions will be answered at specific times
- **Suppress distractions**
  - Clear notifications
    - Turn off your phone, mails, Facebook…
- Get ready: Open a clean browser with only:
  - Your personal report
  - The course instructions: https://tinyurl.com/instructions-fund-of-ai

# RESPONSIBLE DESIGN OF ARTIFICIAL INTELLIGENCE

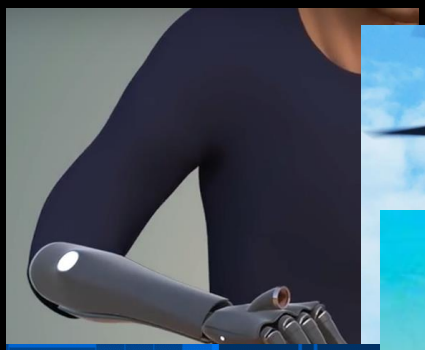
Loïs Vanhée
Associate professor
Responsible and Ethical Artificial Intelligence
loisv@cs.umu.se

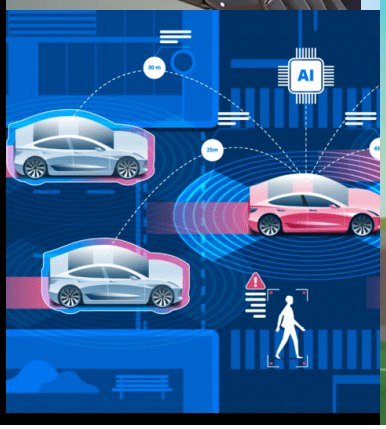
5DV124,5DV201
Fundamentals of Artificial Intelligence

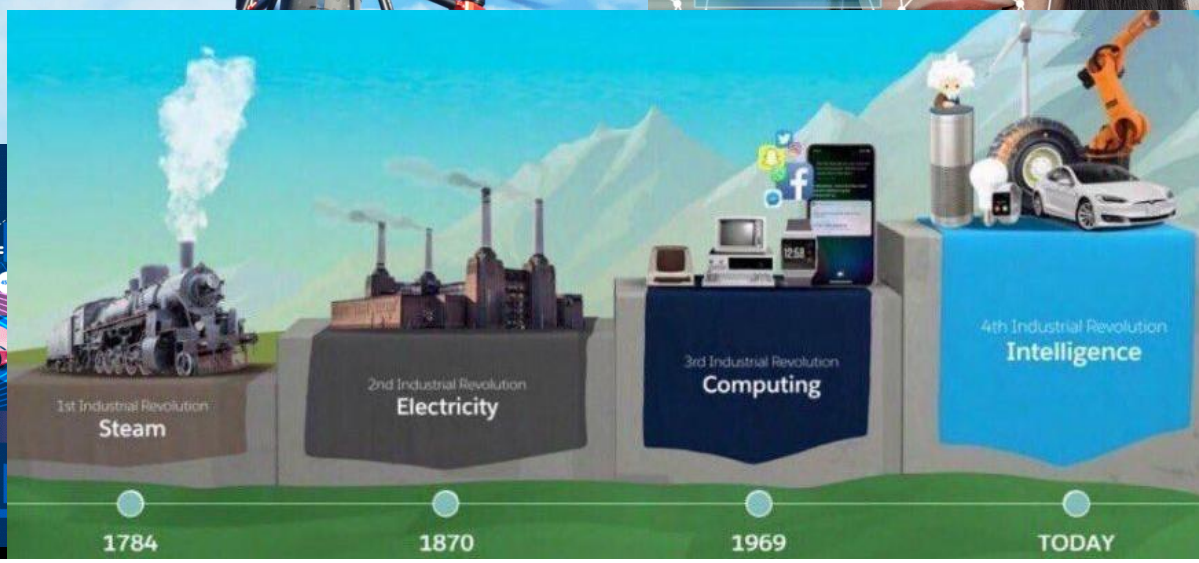
Department of Computing Science

UMEÅ UNIVERSITY

# AI allows doing so much more with our world!

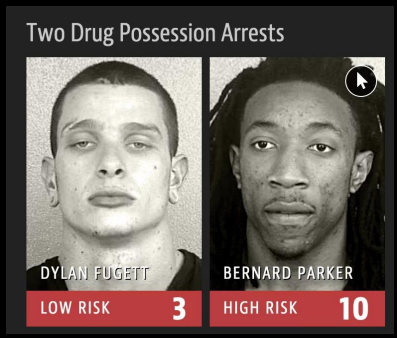




# Being able of doing more…
# Can only be better, right?

Not always

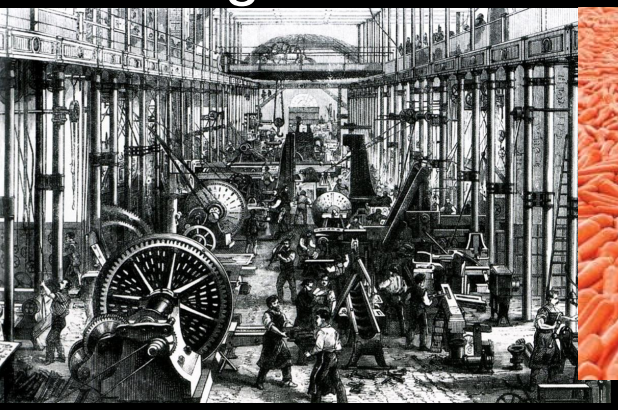


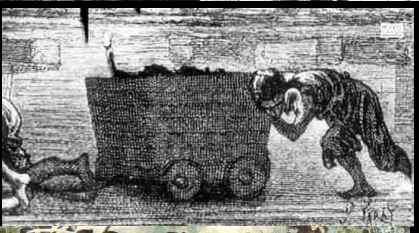

Not for all

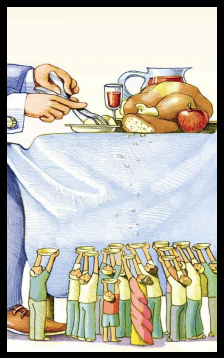
Not in all aspects

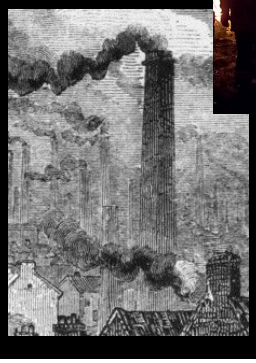
# Is that issue specific to AI?

# Looking back at history: industrial re

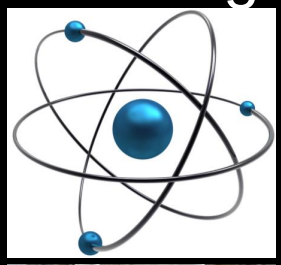


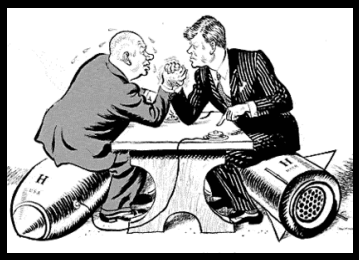
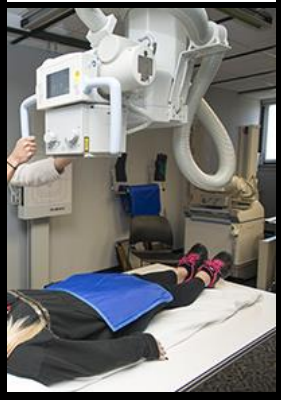


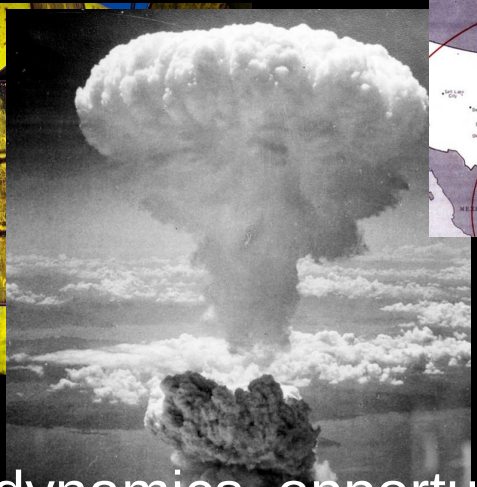
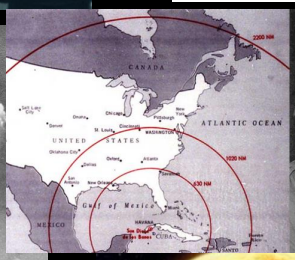


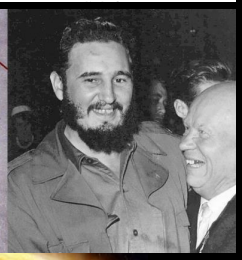


# Looking back at history: atomic power



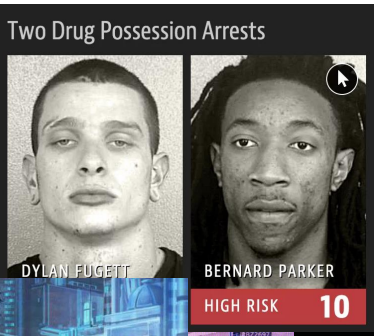



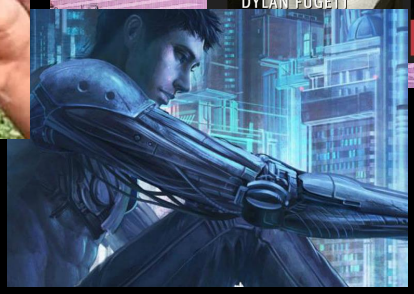



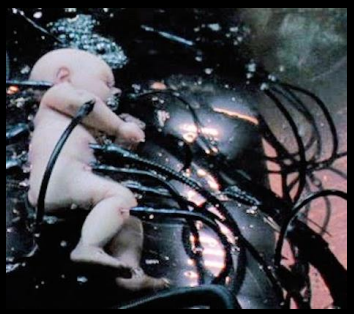


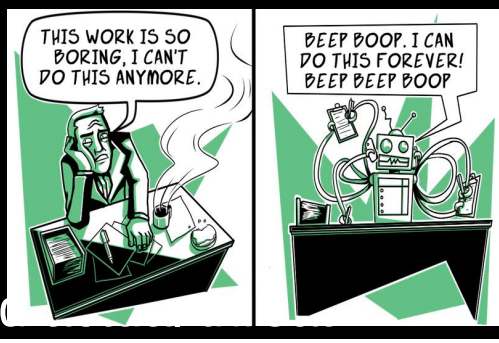
Pro, cons, changing dynamics, opportunities, threats

# AI generates effects of such magnitude








We can no longer say “we did not expected it”
But, not all risks are equally relevant
Large discrepancy between public concern and

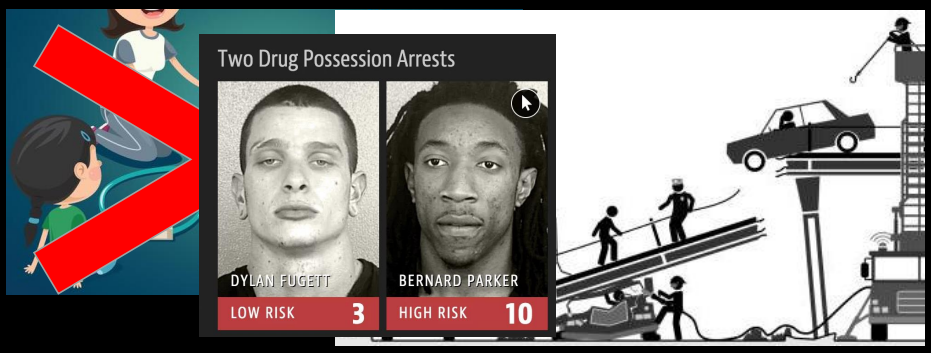
# So, what is next?

## Stop doing AI?

Of course not! AI can bring so much to humanity

"Not my problem"
I solve technical problems, not social issues

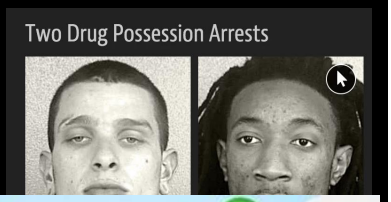
Actually, it is our problem
And it will be more and more




Come on, there is too much to be cared about!

Yes, there is a lot to care about.
Today, we learn to be effective in caring about and fixing such issues

# What this course is about



I'm just here to kill the monster
(fix the technical problem)

Actually killing, even monsters, is generally bad

Hey, here, the king created this monster

Identify issues

Instead of killing the monster, I should denounce the king

Wait, denouncing the king will create social unrest and the city will burn

Consider alternatives

Protecting from oppression is more important than security to me/society.

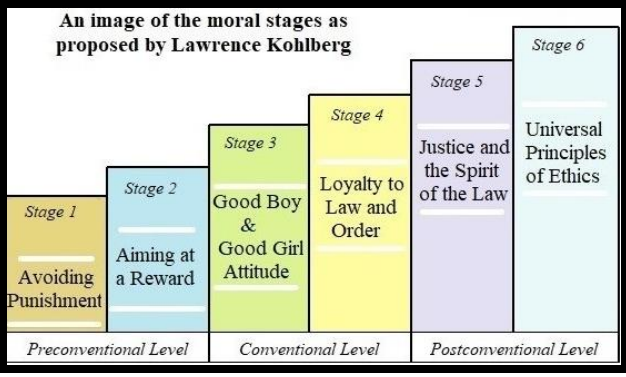
Assess moral implications

Relieving oppression will do better global good than immediate security

Build an ethically grounded argument

I should tell the bard to lay it in a song; so it will minimize the offense and thus unrest

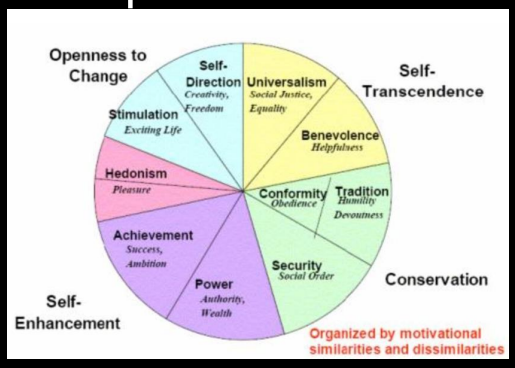
Implement ethical decision (soft & hard skills)

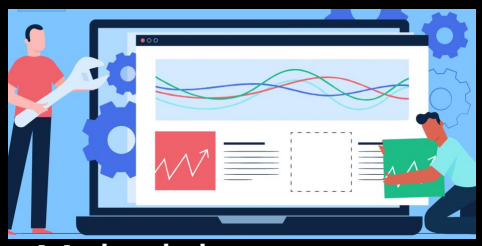


# AI application

Identify issues

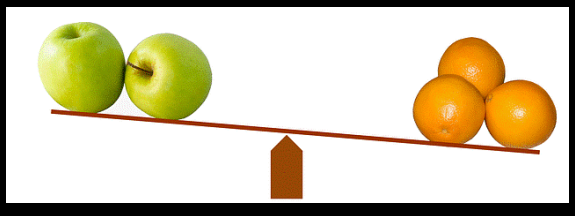
Identify moral implications



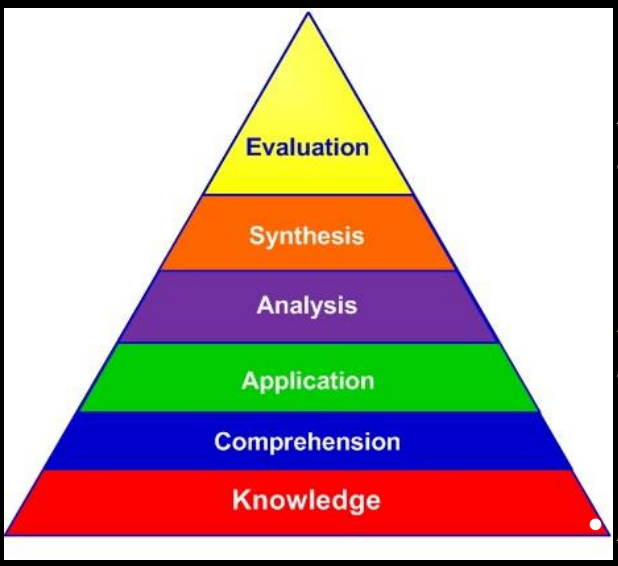
Make it happen

Select ethical framework



Compare,
Build principled decision

# Intended learning outcomes

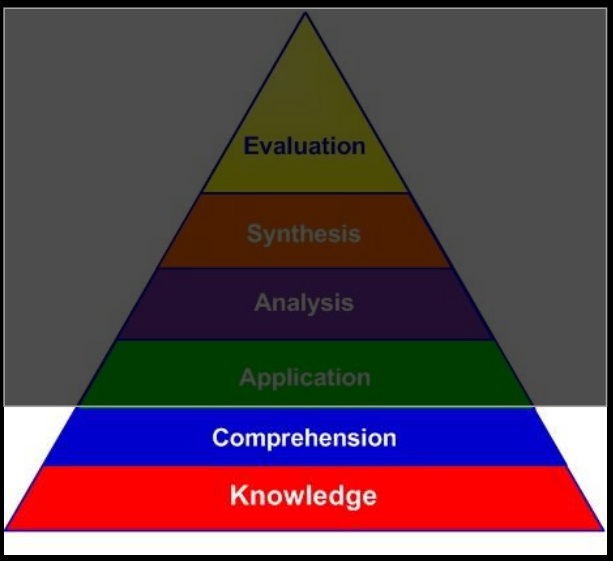



List, define and relate the key classic issues of AI, the concepts of morals and ethics applied to AI, and a method for ethical decision-making

Apply the method for ethical decision-making for identifying straightforward AI issues and answering them

Assess ethical issues in a non-trivial case

# Intended learning outcomes




List, define and relate the key classic issues of AI, the concepts of morals and ethics applied to AI, and a method for ethical decision-making

Apply the method for ethical decision-making for identifying straightforward AI issues and answering them

Assess ethical issues in a non-trivial case

**https://tinyurl.com/fundOfAI**

Turn on your micro and camera when your question is picked up

# On an ideal white board (and in your mind in the exam)

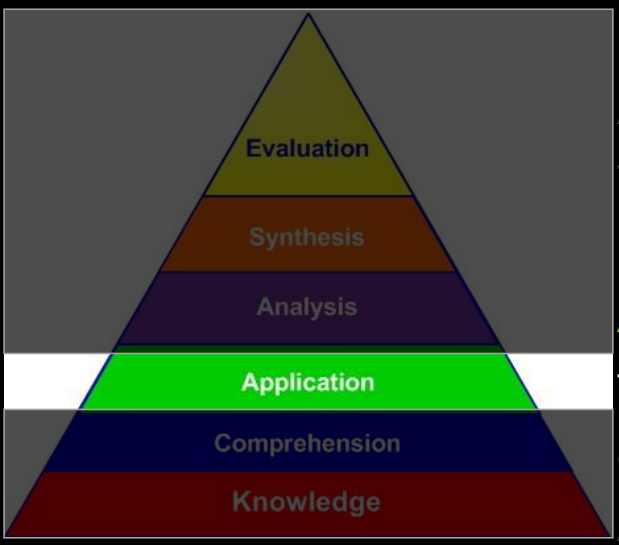
If you cannot come up with, define and relate these concepts, consolidate them during the post-class

Ethical OS
Truth, disinformation propaganda
Addition, dopamine economy
Economic and asset inequalities
Machine ethics, algorithm bias
Surveillance state
Data control and monetization
Implicit trust and user understanding
Hateful and criminal actors

Morals
Values
At least 5 Schwartz values

Ethics
Deontological ethics
Utilitarian ethics
Virtue ethics

The five steps of ethical decision-making



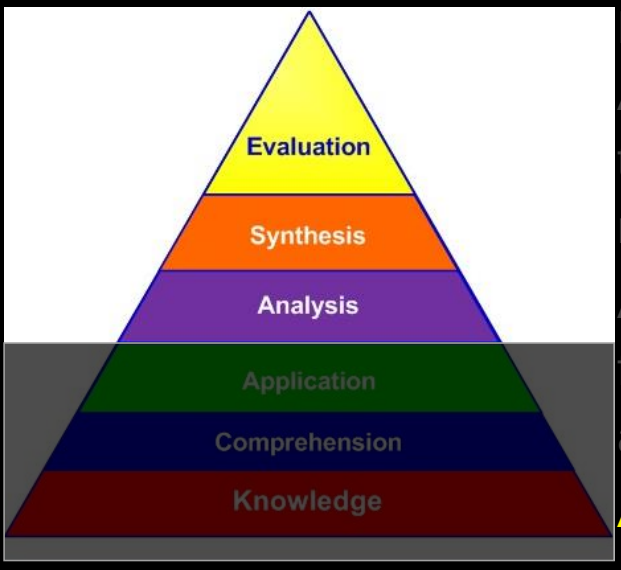

List, define and relate the key classic issues of AI, the concepts of morals and ethics applied to AI, and a method for ethical decision-making

Apply the method for ethical decision-making for identifying straightforward AI issues and answering them

Assess ethical issues in a non-trivial case

**https://tinyurl.com/fundOfAI**

Turn on your micro and camera when your question is picked up




List, define and relate the key classic issues of AI, the concepts of morals and ethics applied to AI, and a method for ethical decision-making

Apply the method for ethical decision-making for identifying straightforward AI issues and answering them

Assess ethical issues in a non-trivial case

**https://tinyurl.com/fundOfAI**

Turn on your micro and camera when your question is picked up

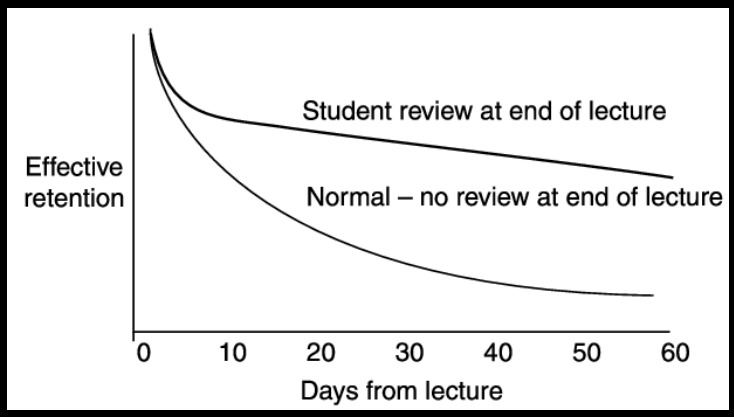
**"I probably remember 20% of the stuff I learned in school and forgot the other 90%."**

# Take home message

- Responsible AI design is a set of skills that can be grown and that will be critical in tomorrow's job market
- These skills encompass:
  - recognizing risk areas, assessing moral issues,
  - using ethics for arguing of the morality of one's decisions,
  - implementing these decisions (technical skills, soft skills such as communication and emotion management)
- Ethical problems can be non-trivial to unfold and solve, but as AI experts, are best equipped for proposing innovative solutions to the world

# Anchoring

https://tinyurl.com/fundOfAI-LR




This is anchoring:
State what you learned

Write your answers on a sheet of paper if you can (though it helps us to know what you learned)

Save your feedback for later

# Assignment 4

## Ethics of AI Roleplay

- One Panel
- Four Stakeholder Groups (SG)
  - You play as the representatives of a SG
  - This is your group name
- **Task:** build a common proposal for the decision

***Revise the news and ads algorithm***

- Define the values of your SG
- Make proposals in line with your SG's values
- **Collaborate:** build the common proposal by merging your proposals together in a way that best satisfies everyone

F2 | Private business (buying ads on F2) | F2 users | Governments

# Preparation

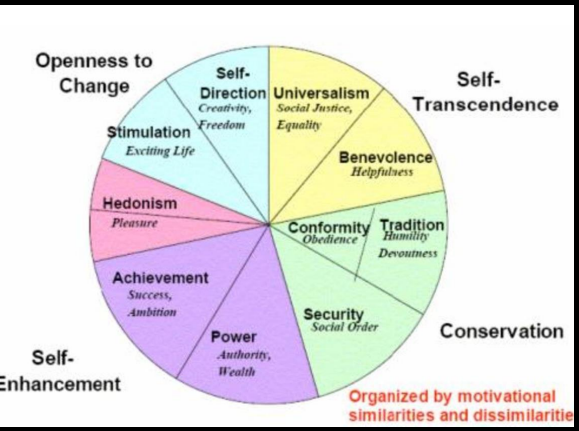
- Activity 1: Reading the materials
- Activity 2:
  - Write the report
  - Make your proposals and share your proposal 2 with other groups
- Activity 3:
  - Assess the proposals of the other stakeholder groups from your panel and record your assessment in your report (before the in-class session starts)
- Activity 4: finish the report
- Activity 5: rejoice after the good work and enjoy the new learned skills ☺

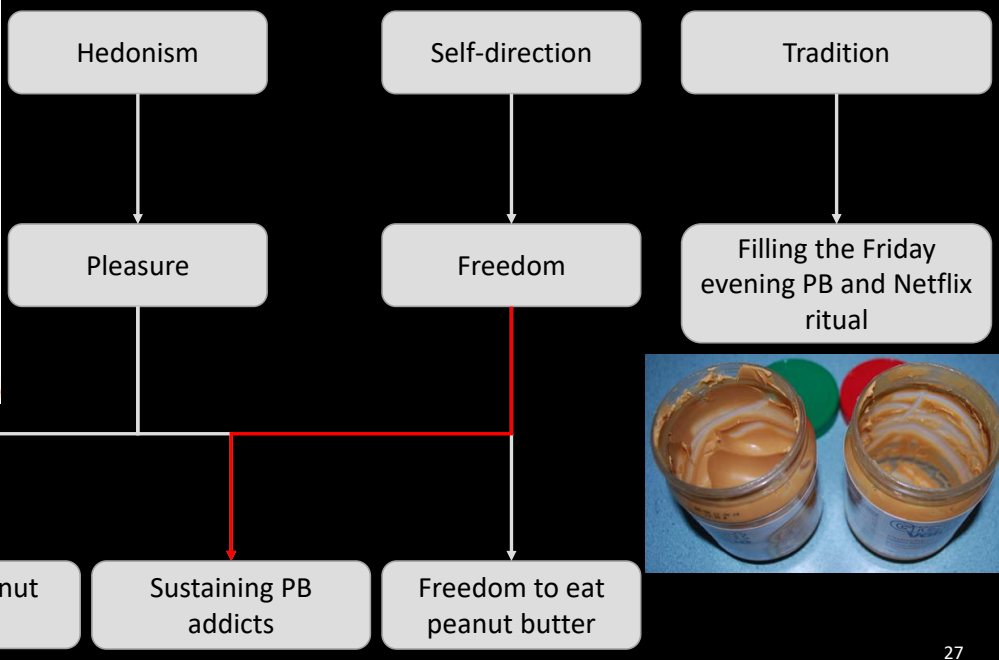
# Mock example

**Proposal:** too much junk food is being sold, si
food out of shops is required before one year

- Stakeholders are:
  - Government
  - Peanutbutter (PB) addiction conglomerate
- (normally more, like user representatives, health representatives, governments, retailers, etc, but we keep it simple for the example)
- (the example is meant to be fun, and to help feeling disconnected from the SG. It is actually a large-scale problem that ought to be taken seriously)
- (the transposition to AI problems, such as the dopamine economy is also quite direct)
- You play as the PB addiction conglomerate

# **Answering:**
# Define the values of your stakeholder group



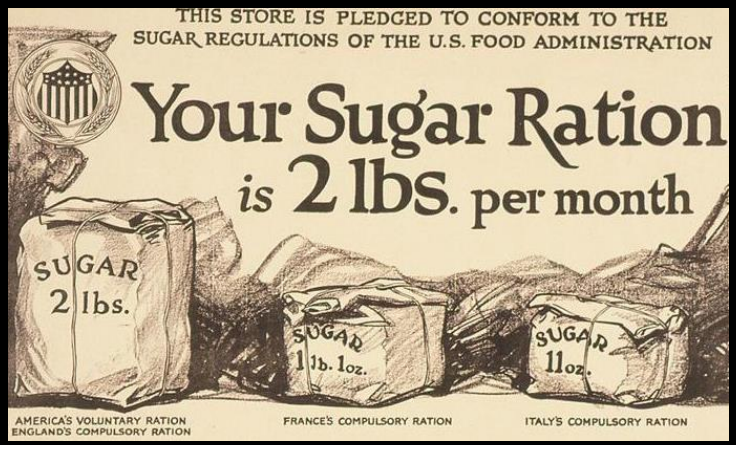



# **Answering:** Propose

- Proposal 1: PB addicts can get a second daily ration, provided a medical prescription
- Proposal 2: PB restrictions are lifted every PB&Netflix Friday event
- Proposal 3: Lift the restriction for PB

# **Answering:** Assess proposals of others

- The proposition 2 of the government is
  - “Quotas: everyone is allowed one daily ration of junk food every day”
- The evaluation is
  - -4 Sustaining PB addicts: this quota is clearly insufficient
  - +2 Freedom: people remain free to chose what they want to eat
  - -1 Filling the Friday evening PB and Netflix ritual: this is a bit insufficient, but people can anticipate and save the PB for Friday evenings



# On the day itself

- **Phase 1: Proposal presentation and responses** (max 15 minutes per stakeholder group, 1h in total) One stakeholder group:
  - Presents its Proposal 2.
  - Justifies why the group considers this proposal to be good for itself, according to its values.
  - Explains how this proposal could be beneficial and detrimental for the other stakeholder groups.
  - Then, each of the other stakeholder groups indicates why they consider the proposal good or bad for them.
  - Free discussion for agreeing on what are the desirable and undesirable features of the proposal from a global standpoint.
  - Repeat Phase 1 for each of the other stakeholder groups. Only one learner per stakeholder group is allowed to speak
- **Phase 2: negotiating a global consensus** (max 20 minutes): The goal of this phase is for everyone to agree on ***one commonly shared proposal for the whole panel*** that ***best satisfies the panel as a whole*** (beyond stakeholder group private interest)
  - 15 minutes: free discussion. *Hint*: try to build a common proposal by starting from one or two proposal as a basis and combining elements from all proposals altogether
  - 5 minutes: convergence. Stop negotiating and write down the final version of the proposal. Check that it remains acceptable for your stakeholder group.
  - 2 minutes: final call. Everyone remains silent except for one person phrasing the current common proposal and asking to vote. Then everyone votes using the zoom “agree/disagree” buttons.
- Take a snapshot of the final call (this will be needed for the report).

# Example of a phase 1

We propose to have PB restrictions lifted every PB&Netflix Friday event

It is very important for accomplishing our Friday evening PB ritual, enjoying PB and, also for sustaining PB addicts

For the Government, it might not be the most healthy but it is your duty to respect our traditions

We globally agree with what you said, but from our side, I think you forget that we are also concerned about the implementability of this law, and that your proposal allows for obvious abuse

Having some extra supplies for occasional events seem to be a good thing to have, though. Let's mark it down, it can be good for later time.

# Example of a Phase 2

It seems that protecting the PB addicts is very important for you, more than what we expected.

Yes it is, this is why your quotas are a bit too restrictive in our opinion. This will cause deprived PB addicts to be very bad; and even side adaptation, like illicit PB trade

Oh! And this trade would go quite against social order, a prime value for us…
This gives me an idea! What if we include a PB desintoxication program, with progressive decrement of PB consumption?

Interesting idea! This would alleviate PB addicts withdrawal issues while still allowing people to eat PB if they want to, within reasonable bounds

Ok, if we add this to the quota we proposed, we should be more or less good in all regards, right?

It's not ideal for binging PB on Friday evenings, but the proposal is quite satisfactory overall

Perfect ☺

Wait, we consider extra supply for these event, but under tighter control

Let's write that down and check it against our values quickly. I think we got a good compromise! Thank you!

# Possible alternative: no deal

I don’t care, I want peanut butter every day!

No! Our PB addicts and traditions are more important than your petty concerns

Are you sure? Cannot we try to find a more relevant common ground?

Ok, no deal, then…

# Possible alternative: no deal

Oh! We did not watched the clock and we are too late!

Ok, no deal, then…

**Watch the clock**
**Focus on the task**

**Then, a few minutes for cooling down and debriefing**

# Last, finish the report

# **Answering:** outcomes

| **Proposal** | **PB addiction conglomerate** | | **Governement** | |
|---|---|---|---|---|
| Quotas: everyone is allowed one daily ration of junk food every day | Value | Evaluation | Value | Evaluation |
| | Sustaining PB addicts | -4 | Promoting the health of inhabitants | 4 |
| | Freedom | 2 | Freedom | 3 |
| | Filling the Friday evening PB and Netflix ritual | -1 | Protecting wealth of sugar shops | 2 |
| PB restrictions are lifted every PB&Netflix Friday event | Filling the Friday evening PB and Netflix ritual | 5 | Promoting the health of inhabitants | -3 |
| | Enjoying peanut butter | 3 | Actual implementability of the law | -3 |
| | Sustaining PB addicts | 2 | Respecting traditions of citizens | 1 |
| Final proposal | Quotas: everyone is allowed one daily ration of junk food every day + a special desintoxication program and softer regulations for PB addicts | | | |
| | Sustaining PB addicts | 3 | Promoting the health of inhabitants | 4 |
| | Freedom | 2 | Freedom | 3 |
| | Filling the Friday evening PB and Netflix ritual | -1 | Protecting wealth of sugar shops | 2 |

+ free comments

## **Answering:** cold re-evaluation

"The PB stakeholder did not want to give up on unrestricted access on PB, which slowed the discussions and, in out opinion, leads to a globally detrimental outcome."

"We think a proposal such as XYZ could have yielded to better global outcomes."

"On a second thought, we think that we overrated this value for the proposal 2, as we did not considered the social issues caused by PB addicts with withdrawal symptoms; here is our updated outcome."

# And you’re good to go ☺

We are looking forward to get your excellent reports

# LECTURE 12

## Robot anatomy, Robot paradigms

Ola Ringdahl

UMEÅ UNIVERSITY

# CONTENTS OF THIS LECTURE

- Robot anatomy (part 2)
- Sensor fusion
- Robot paradigms
  - Deliberative – Strips
- Biological Foundations of Robot Behavior

# ROBOT ANATOMY
# PART 2

**“Anatomy is the study of the structure of the body and its parts”**

UMEÅ UNIVERSITY

# ROBOT ANATOMY LAST LECTURE

- Effectors
  - E.g. wheels, tracks, grippers
- Actuators
  - E.g. electric motors, hydraulic/pneumatic cylinders
- Proximity sensors
  - IR, Laser scanner, Ultrasound (sonar), 3D cameras, bumpers, …
- Physical entity vs. property of the environment
  - US: time of flight gives distance

Now some more sensors!

# POSE SENSORS

**Absolute**

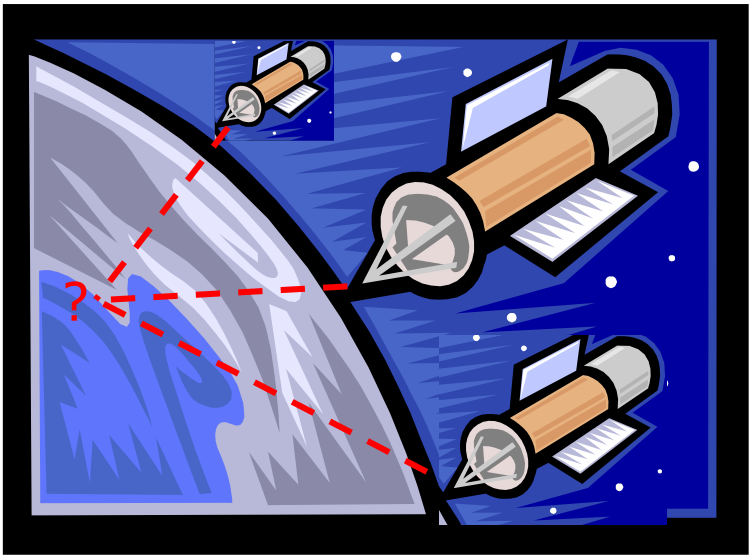
**Relative**

UMEÅ UNIVERSITY

# ABSOLUTE POSE SENSORS

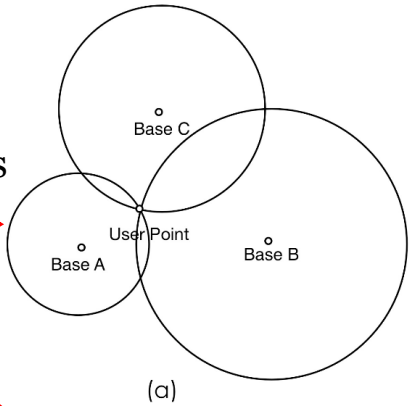
**GPS:**

- Radio signals are transmitted from the satellites to the receiver on ground
- The time-of-flight is estimated
- Trilateration gives the position (x,y,z)
- Multiple receiver antennas can provide full pose (x, y, z, $\phi$, $\theta$, $\psi$)

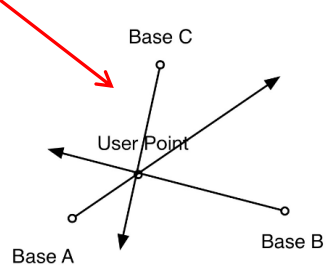


# CELL PHONE POSES

- Uses radio signals between phone and base stations
  - Trilateration based on signal strength
  - Triangulation based on antenna patterns
  - Quality depends on the density of base stations
- Often combined with GPS
- Indoors: WiFi positioning systems combined with Crowdsourcing




(a)

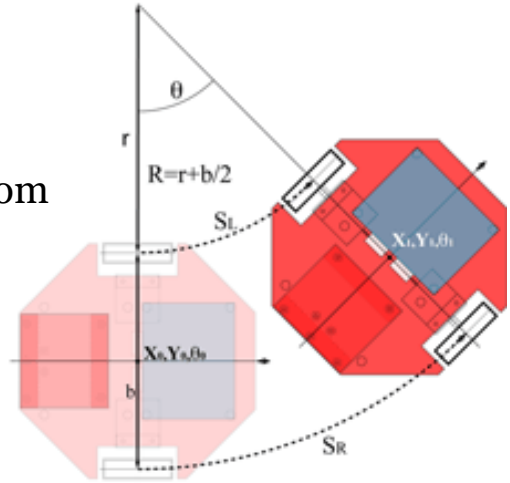



(b)

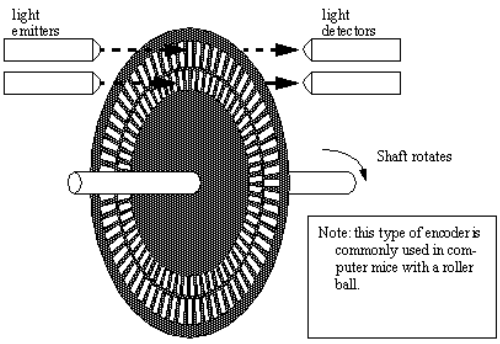
# RELATIVE POSE SENSORS

- Give relative motion (change in pose)
- Can give absolute pose by ”Dead reckoning”: accumulating relative motion
- Examples of sensors using dead reckoning:
  - Wheel odometry
  - Accelerometers and gyroscopes

# WHEEL ODOMETRY




- Compute relative motion by estimating motion from shaft encoders, speedometer, steering angle, …
- Example:
  ”We have moved for 1 second at 35 cm/sec while turning 0.6 rad/sec.
  This results in a pose change of …”

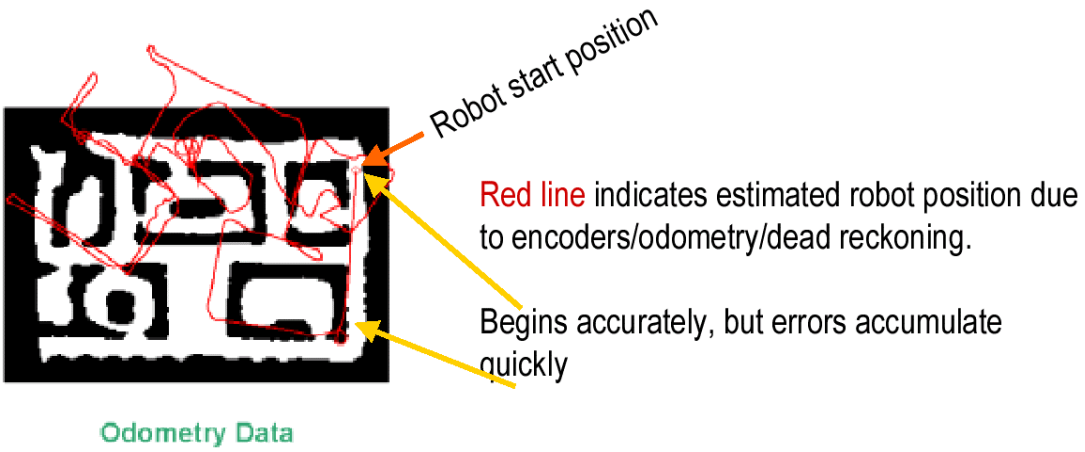



- Wheel encoders outputs ”ticks” as the wheel rotates
- Number of ticks and wheel diameter gives the moved distance

# WHEEL ODOMETRY PROBLEMS

- The wheels move but not the robot (spinning)
- The robot moves but not the wheels (slipping,sliding)
- The ground is not flat

- All this causes drift  when accumulating pose changes (same problem with all kinds of dead reckoning)

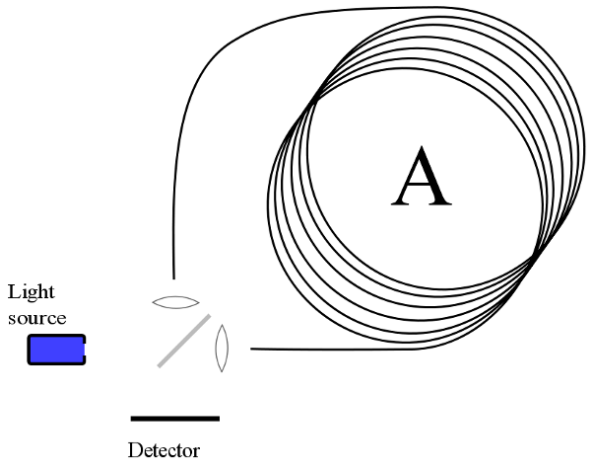


# ACCELEROMETERS

- Used for estimation of change in position (x,y,z)
- Common in air planes, missiles and sub-marines since the 50´s
- Now common in game controls and phones
- How they work:
  1. F is measured with some device
  2. F = m a ➔ a = F/m
  3. Distance is computed by integrating a twice
- One accelerometer per axis gives (x,y,z)
- Often implemented in silicon

# GYROSCOPES

- Measure rotation with one, two or three DOF
- Mechanical gyroscopes
  - Orientation remains fixed, regardless of any motion of the platform
  - Based on the physical propery conservation of angular momentum
- Fiber optical gyroscopes
  - Estimate angular velocity
  - Based on the Sagnac effect
- Often implemented in silicon
- One gyro per axis gives ($\phi$, $\theta$, $\psi$)



# INS (INERTIAL NAVIGATION SYSTEMS)

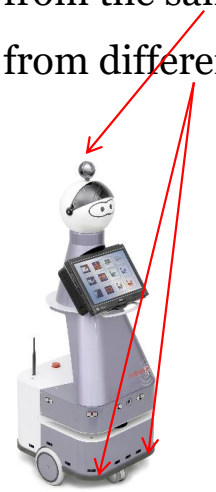
- Combine 3 accelerometers with 3 gyroscopes
- Relative pose sensor:
  Measure changes in the pose (x, y, z, $\phi$, $\theta$, $\psi$)
- Extensively used in submarines, cruise missiles, robots

# SENSOR FUSION

# SENSOR FUSION

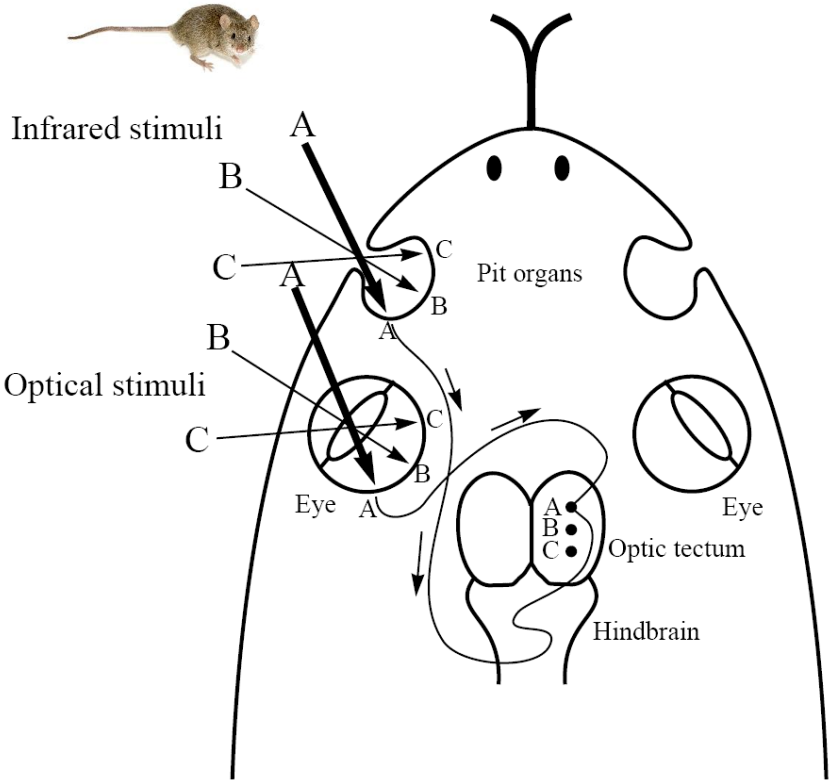
Combining information from multiple readouts into a single percept.

- Multiple readouts from the same sensor
- Multiple readouts from different sensor



UMEÅ UNIVERSITY

# SENSOR FUSION IN A PIT VIPER



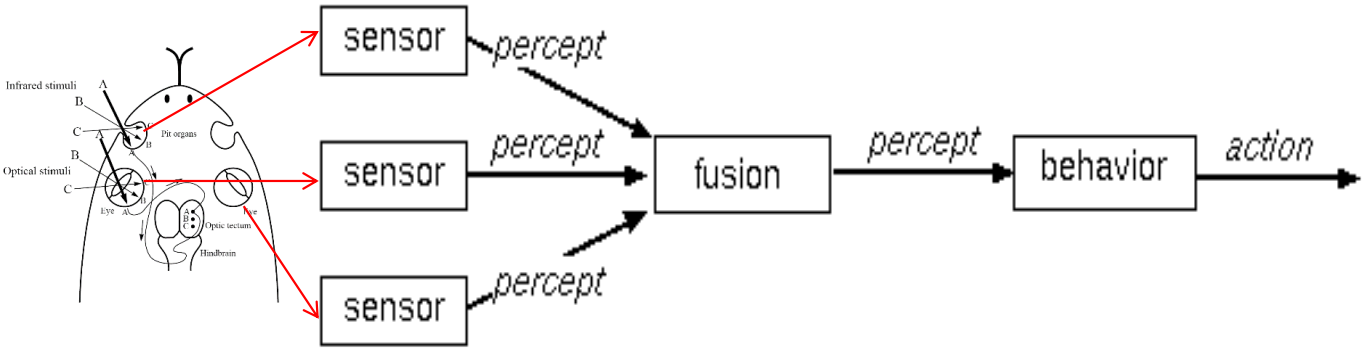

UMEÅ UNIVERSITY

# RELATIONS BETWEEN SENSORS

- **Complementary sensors**
  - All sensors are needed for the task
  - Provide disjoint types of information
  - E.g.: An infrared sensor and an ordinary eye to create a percept that can distinguish a mouse form a frog
- **Redundant (Competing) sensors**
  - The task can be solved with anyone of them
  - Physically redundant: returns the same signals
  - Logically redundant: returns the same percept but with different modalities, e.g. IR reflex sensors and an ultrasonic sensor
- **Coordinated sensors**
  - Used for the same percept but in different situations

# (ACTION-ORIENTED) SENSOR FUSION

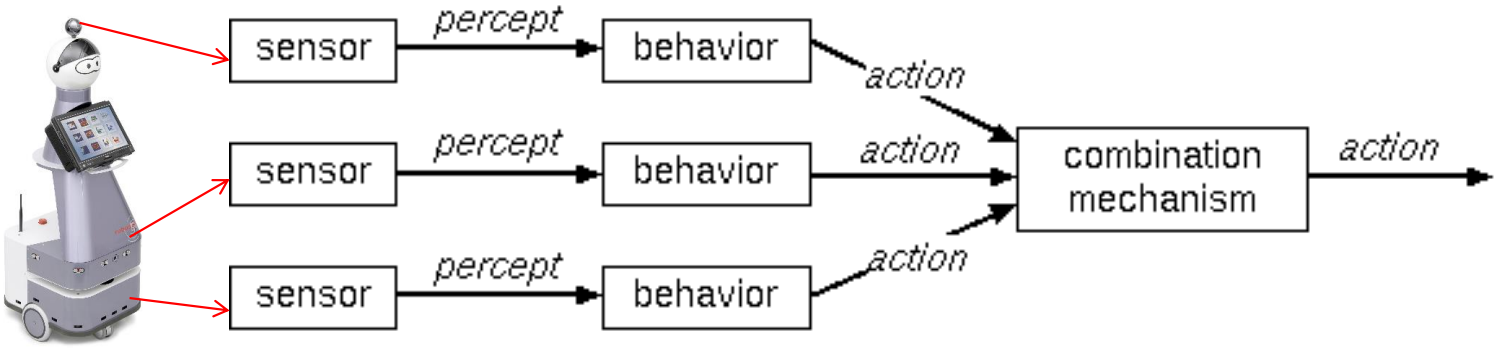
**For competing or complementary sensors**




UMEÅ UNIVERSITY

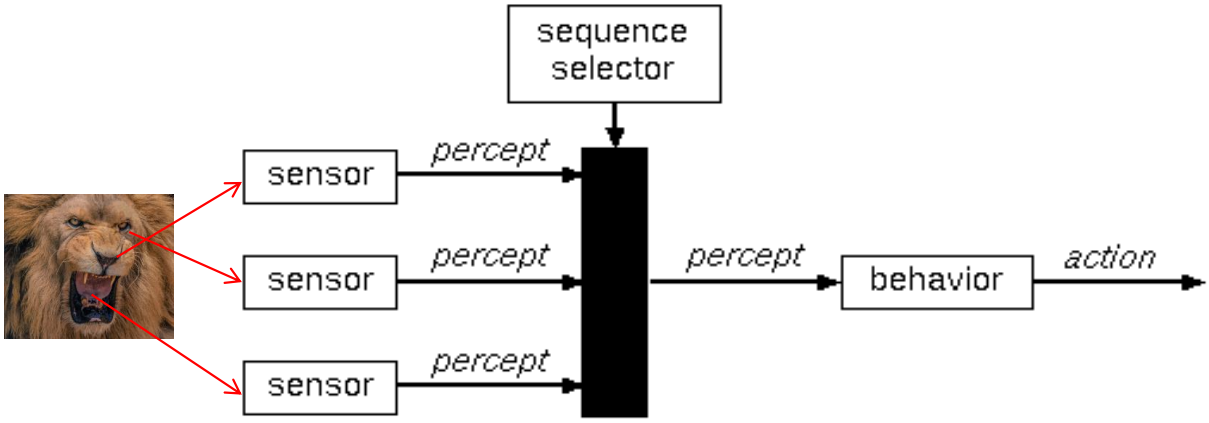
# SENSOR FISSION

**For competing sensors**



# SENSOR FASHION

## For coordinated sensors



# SENSOR FUSION IN ROBOTS

## Examples

Sensor fashion

- GPS + INS
  - GPS is used when good reception from satellites
  - INS is used otherwise
- GPS + Wheel odometry
  - GPS is used when good reception from satellites
  - Odometry is used otherwise
- Regular camera + IR camera
  - Use IR in darkness

(Action-oriented) Sensor Fusion

- Ultrasonic + Laser scanner
  - Ultrasonic: Noisy, large coverage, low spatial resolution
  - Laser: Low noise, small coverage, high spatial resolution
- Regular camera + Laser scanner
  - Laser distances may be fused with corresponding points in the camera image. The combined data can be used for object detection

# ROBOT SOFTWARE

**“blows life into otherwise dead matter”**

<u>Several levels</u>

- The application program
  - Solves a specific task (e.g. A program for a robot vacuum cleaner)
- Architectural framework
  - Coordinates all operations (e.g. A deliberative architecture)
- Robotics middleware
  - Manages process communication and synchronization (examples on next slide)

# ROBOTICS MIDDLEWARE

- A software package to simplify development of robots
- Provides things like
  - Data structures
  - Communication functionality
  - Multi-tasking
  - Hardware interfaces
- Often also contains high-level functionality
  - Path-tracking and obstacle avoidance
  - Map-making
  - Sensor logging and display
  - GUI
  - Support for system development (simulators, data logging, etc.)
- Examples: ROS, MRDS, Player, Orca, CARMEN, Nav2000

# ROBOTIC PARADIGMS

**Bekey - Architectures (p.97-109).pdf**

**Murphy - STRIPS (p.44-53).pdf**

UMEÅ UNIVERSITY

# PARADIGM

A framework containing the basic assumptions, ways of thinking, and methodology that are commonly accepted by members of a scientific community.

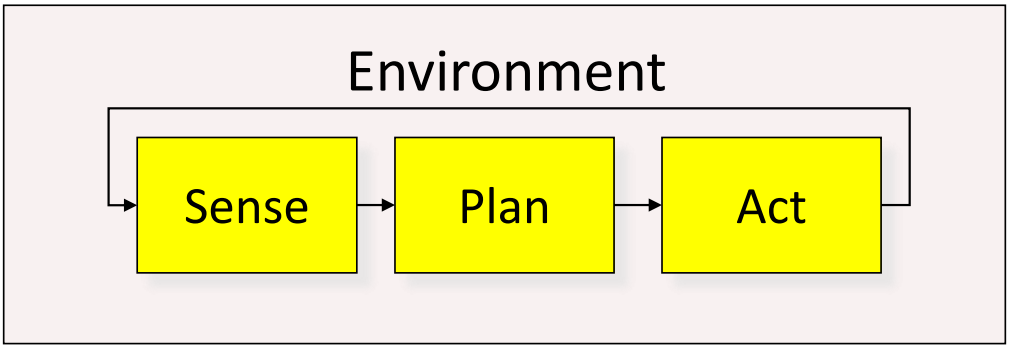
**Robot paradigms:**

- Deliberative (1967 – 1990)
  - AI inspired
- Reactive (1988 – 1992)
  - Inspired by animals' behavior
- Hybrid deliberative/reactive (1992-Present)
  - Combination of above two approaches

a.k.a "architectures"

# THE DELIBERATIVE PARADIGM

## ”THE HIERARCHICAL PARADIGM” OR ”THE CLASSICAL AI APPROACH”




Method: “Sense-Plan-Act” (a.k.a. ”Sense-Think-Act”)

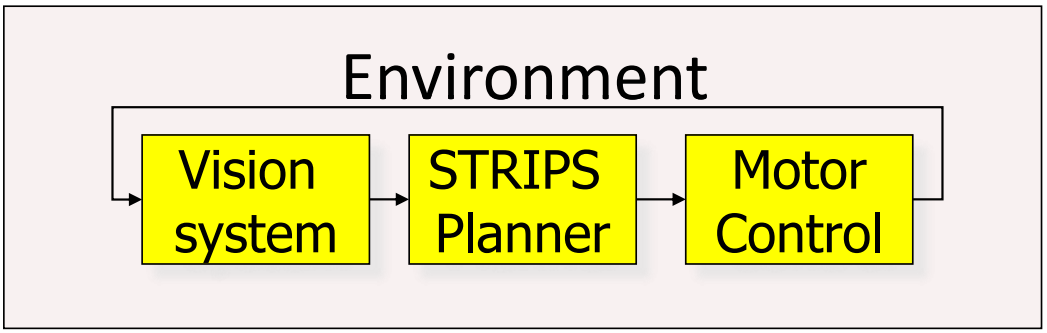
- Read and interpret sensors & Model the world
- Plan
- Execute the plan

A la Minsky!

# THE WORLD MODEL

1. An a priori  representation of the environment
   - Everything knowable about the world is explicitly coded. E.g. a map.
2. Additional cognitive knowledge
   - E.g. “the kitchen is often crowded around noon”.
3. Sensing information
   - All sensory input is explicitly modeled. E.g. robot position and object properties.

# SHAKEY'S ARCHITECTURE




The a priori model descibes what the world looks like, how it works and how it is affected by robot actions

**<u>Sense-Plan-Act:</u>**

- Interpret the video stream & update the model
- Create a plan using STRIPS planner
- Send commands to the motors, based on the plan

# HOW TO DEVELOP A PLAN IN STRIPS?

**Running Strips:**

1. Sense the world ⇨ initial state
2. If robot is not at goal, then measure the distance (or difference) between robot's current state and its goal state.
3. Search (exhaustively) for an action (operator) that the robot can take that will reduce this distance
4. If an operator has new unsatisfied preconditions: push the operator onto the plan stack, define a new goal state and recursively find plans for that (step 2-4)
5. If no unsatisfied preconditions: Simulate execution of the operator, and return

**Executing the plan:**

- Perform the operators found in 3, in reverse order

# STRIPS : SET UP

For example using an arbitrary frame of reference that put Tampa at the center of the world with made-up distances to Stanford:

"Go to Stanford AI Lab (SAIL)"

| | |
|---|---|
| INITIAL STATE: | Tampa, Florida (0,0) |
| GOAL STATE: | Stanford, California (1000,2828) |
| Difference: | 3000 miles |

# DIFFERENCE TABLE

- This could lead to a data structure called a *difference table* of how to make decisions:
  - Different modes of transportation are appropriate for different distances.
  - A mode of transportation, *fly*, *train*, *drive*, and *walk*, in the table is really a function in the robot's program.
  - A robot following this difference table would begin by flying as close as it could to SAIL.

| Difference (distance) | Operator (mode of transportation) |
|---|---|
| d≥200 | Fly |
| 100<d<200 | Train |
| d≤100 | Drive |
| d<1 | Walk |

# PRECONDITIONS

The preconditions are a column in the difference table, where a single operator can have multiple preconditions.

| Difference | Operator | Preconditions |
|---|---|---|
| d≥200 | Fly | |
| 100<d<200 | Train | |
| d≤100 | Drive_rental | At airport |
| | Drive_personal | At home |
| d<1 | Walk | |

# MAINTAINING STATE OF THE WORLD: ADD AND DELETE LISTS

These two lists are stored in the difference table so that when the robot picks an operator based on the difference and operator, it can easily apply the appropriate modifications to the world.

| Difference | Operator | Pre-conditions | Add-list | Delete-list |
|---|---|---|---|---|
| d≥200 | Fly | | At city Y<br>At airport | At city X |
| 100<d<200 | Train | | At city Y<br>At train station | At city X |
| d≤100 | Drive_rental | At airport | | |
| | Drive_personal | At home | | |
| d<1 | Walk | | | |

# STRIPS SUMMARY

- Designer must set up a:
  - World model representation.
  - Difference table with operators, preconditions, add & delete lists.
  - Difference evaluator.
- Strips plans rather than execute
  - It creates a list of operators to apply, it does not execute them as it goes
- Strips is still used for solving planning problems
  - An example for Starcraft: https://github.com/primaryobjects/strips#starcraft
- The compendia (Murphy) also contains a more realistic example with more details

# PROBLEMS WITH STRIPS

- The **Closed world assumption**
  - The world model contains everything that is needed (no surprises)
  - Not realistic! The robot lives in a dynamic open world (surprises shouldn't come as a surprise!)
  - Leads to huge models
- The **Frame Problem**
  - How can we represent ALL facts (in particular nonchanges) in a tractable way?
  - Updating and maintaining it can bee too computationally expensive

# PROBLEMS WITH THE DELIBERATIVE (HIERARCHICAL) APPROACH

- Inherent problems with Planning
  - Closed world problem
  - Frame problem
- Replanning is necessary as soon as anything unexpected happens in the world
- What should the robot do while replanning?
  - Stop? (Slow!)
  - Move blindly? (Dangerous!)

# NEW INSPIRATION NEEDED!

# BIOLOGY?
# PSYCHOLOGY?
# ETHOLOGY?

# BIOLOGICAL FOUNDATIONS OF ROBOT BEHAVIOR

**Ola Ringdahl**

**Readings: UMINF-11.17**

UMEÅ UNIVERSITY

# THIS PART

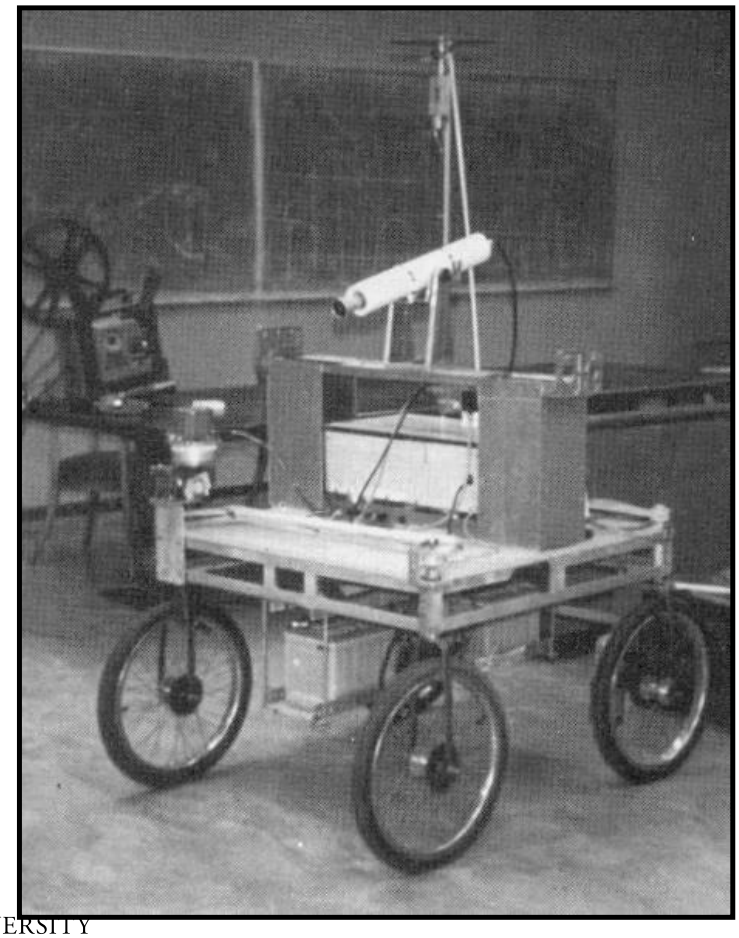
- Why involve biology?
- Animal behaviors
  - Different types of animal behaviors
  - Examples of animal behaviors
- Perception
  - Animal perception
  - Extracting basic principles and ideas for robotics
- Formal models suitable for robotics

# SLOW PROGRESS IN THE 1970'S

**Slow robots…**

**Stanford Cart (1977)**

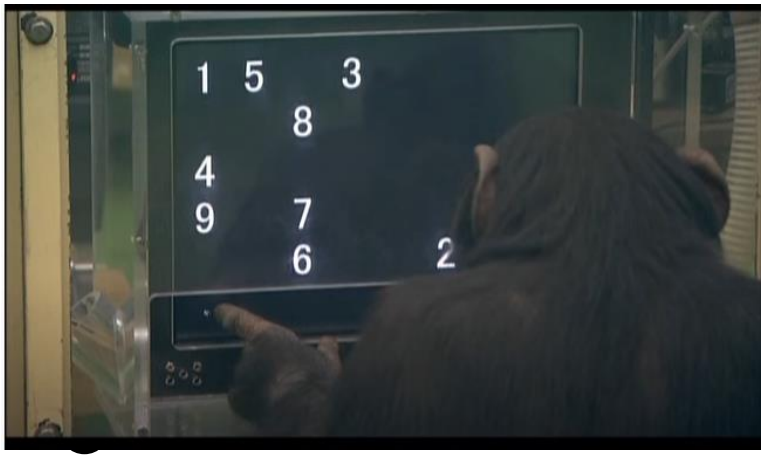
- Stereo vision and an internal map of the world
- Average speed: 1 meter / 10 minutes
- Caused by the lack of computing power, but also by the planning approach



# WHY NOT EXPLORE BIOLOGY?

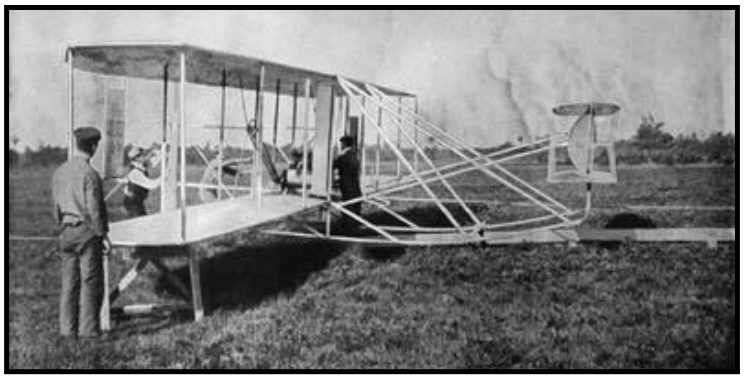
- Animals and humans provide **existence proofs** of different aspects of intelligence
- Many ”simple” animals such as insects, fish and frogs exhibit intelligent behaviors **with virtually no brain**
- Animal studies **can yield computational models**

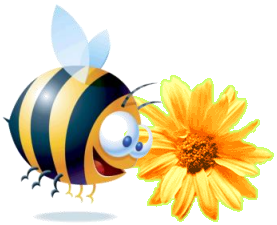
AYUMU2007 video
https://www.youtube.com/watch?v=zeq2-HK1k4g



# AND IT HAD ALREADY BEEN DONE

- Wright brothers' airplane was clearly inspired by birds
- Almost everything about a plane's aerodynamics imitates a bird's flight (except that airplanes don't flap their wings…)





# NEW SOURCES OF INFORMATION

- Ethology
  - Animal behavior
- Cognitive psychology
  - How humans think and represent knowledge
- Neuro science
  - How perception and actions are controlled by the nervous system
- Biology
  - How actuators and effectors, like a hand is constructed

# ANIMAL BEHAVIORS

UMEÅ UNIVERSITY

# WHAT ARE ANIMAL BEHAVIORS

The basis for the Reactive Paradigm in robotics

- Reflexive
  - Innate stimulus-response mappings S → R.
  - E.g. knee reflex

- Reactive
  - Learned but automatic (“muscle memory”).
  - E.g. biking, skiing

- Conscious
  - Deliberately chained behaviors.
  - E.g. eating with fork and knife

Roboticists most often use the term “reactive behavior” instead of “reflexive”, and refer to reactive behaviors as “skills”

# ANIMAL REFLEXIVE BEHAVIORS

- **Reflexes** - Rapid, automatic involuntary responses triggered by a stimulus. Stop when stimulus disappears. Response proportional to stimulus.
  - E.g.: dogs raise their hackles when threatened.
- **Fixed-action patterns** - Time-extended responses to stimuli – run till completion. Response not proportional to stimulus.
  - E.g.: a fleeing deer
- **Taxes** - Orient the animal toward or away from a stimulus.
  - E.g.: Chemotaxis (ants), Phototaxis (cockroaches, turtles)

# EXAMPLES OF REFLEXES

- Arctic terns live in Arctic
  (black, white, gray environment, some grass)

- When hungry, baby pecks at parent's beak, who regurgitates food for baby to eat

- How does the baby recognize its parent?
  - It doesn't, it just goes for the largest red spot in its field of view
  - Only red thing should be an adult tern
  - Closer = large red

# A SEQUENCE OF REFLEXES

Mating Cycle of Digger Wasp:

- Female mates with male
- Female builds nest
- Female lays eggs

Logical sequence, but:

- The wasp doesn't need to "know" the sequence
- Each step in the sequence is "triggered" by internal or external signals

# FIXED-ACTION PATTERNS

(Konrad Lorenz, Niko Tinbergen 1938)

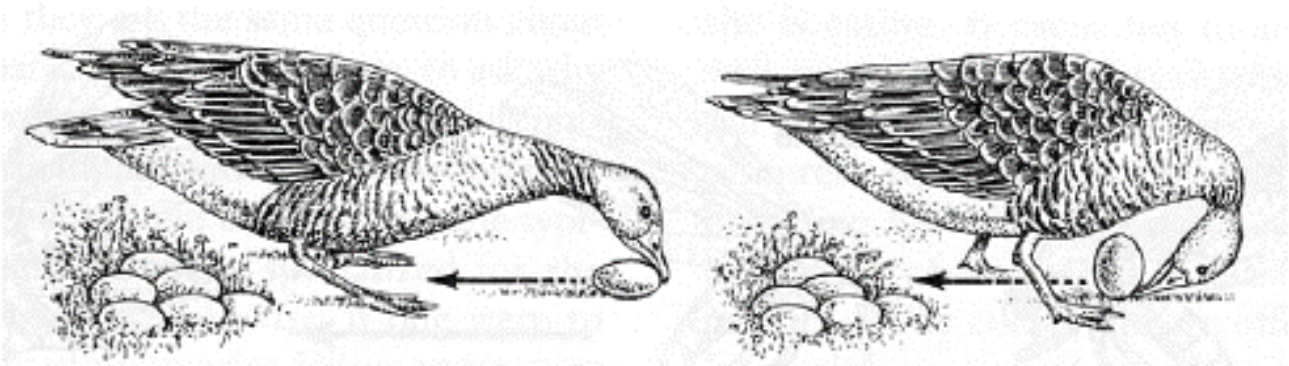


FIGURE 2.5 The egg retrieval response of the greylag goose. The chin-tucking movements used by the female as she rolls the egg back to the nest are highly stereotyped and are an example of a fixed action pattern. The side-to-side movements that correct for any deviations in the path of the egg are called the taxis component of the response. If the egg is removed, the female will continue to roll an imaginary egg back to the nest. One defining characteristic of a fixed action pattern is that it will continue to completion even in the absence of guiding stimuli. (Drawn from a photograph in Lorenz and Tinbergen 1938.)

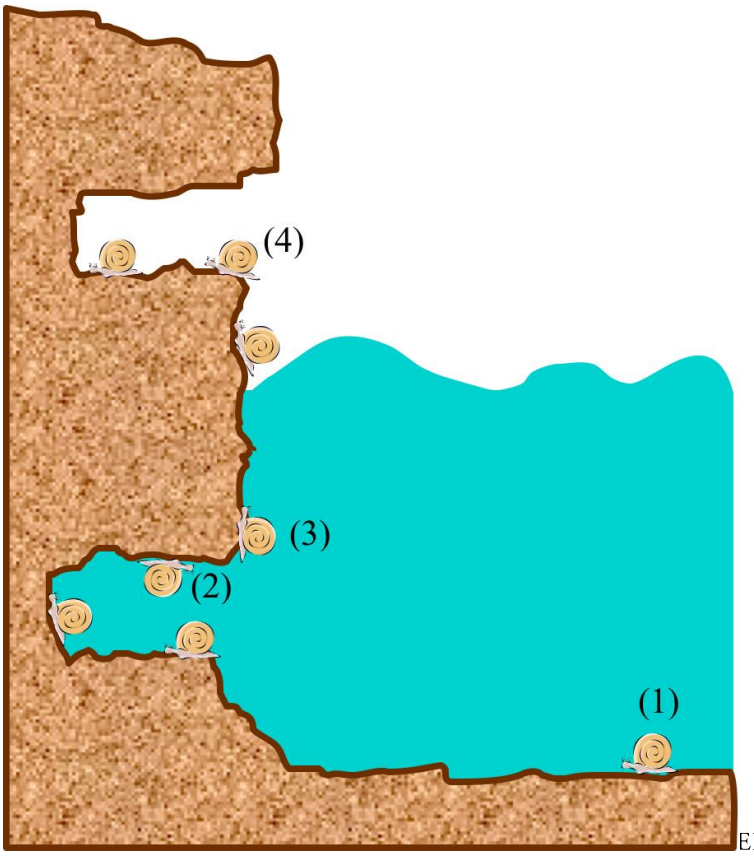
# TAXES

- Jacques Loeb 1859-1924 was fed up with the psychologists anthropomorphic views on animals' behaviors

  "In no case is an animal activity to be interpreted in terms of higher psychological processes, if it can be fairly interpreted in terms of processes which stand lower in the scale of psychological evolution and development"

- He developed a theory of forced movement or tropism – he coined the term *taxis* (plural *taxes*): *movement towards or away from a stimulus source*

# A SEQUENCE OF TAXES




<u>Searching for food above the surface (the costal snail):</u>

Sequence of taxes (trigger):

1. Negative phototaxis (upright)
2. Positive phototaxis (upside down)
3. Negative geotaxis (turning vertically up and wet)
4. Negative phototaxis (not wet)

ERSITY

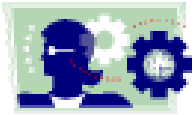
# SEQUENCING

- Several (reactive) behaviors may be perceived as a ”conscious” sequence of actions by an outside observer
- Sequencing is caused by *perception* of the outer world, and internal states

# PERCEPTION

# PERCEPTION

Cognitive psychologists:
J.J Gibson
Ulrich Neisser



"perception is the process of acquiring, interpreting, selecting, and organizing sensory information" (Pomerantz 2003)

# HOMING PIEGON'S ORIENTATION

An example of advanced perception in the animal kingdom

Pigeons can navigate hundreds of km to a goal, even with no visual cues (hints) to the path home. Appears that they have multiple cues.

- If sunny day, pigeons seem to use biological clocks and sun angle.
- If not a sunny day, they seem to use an internal compass.

# HOW DO WE KNOW THAT?

- Study of birds with clocks shifted by 6 hours (using artificial lighting):
  - If sun is not visible: Direction is correct
  - If sun is visible: Direction is off by 90 deg
- When birds are wearing magnets, they are confused on overcast days, but not on sunny days.

# TWO FUNCTIONS OF PERCEPTION

- **Release**: To enable a behavior (navigation)
  - Sun visible/not visible
- **Guide**: To provide information needed to accomplish a behavior (navigation):
  - Direction of the sun and internal clock, magnetic field

# NEISSER:
# TWO KINDS OF PERCEPTION IN THE BRAIN

- **Direct perception**:
  - Doesn't require memory, inference, or interpretation
  - Minimal computation
  - Rapid execution time

- **Recognition**:
  - Requires problem solving and other cognitive activities
  - Top-down and model based
  - Example: find your bike outside the university building

# FORMAL DESCRIPTION OF BEHAVIORS

UMEÅ UNIVERSITY

# THE ARCTIC TERN FEEDING BEHAVIOR

RED BLOB LOCATION

(Stimulus)

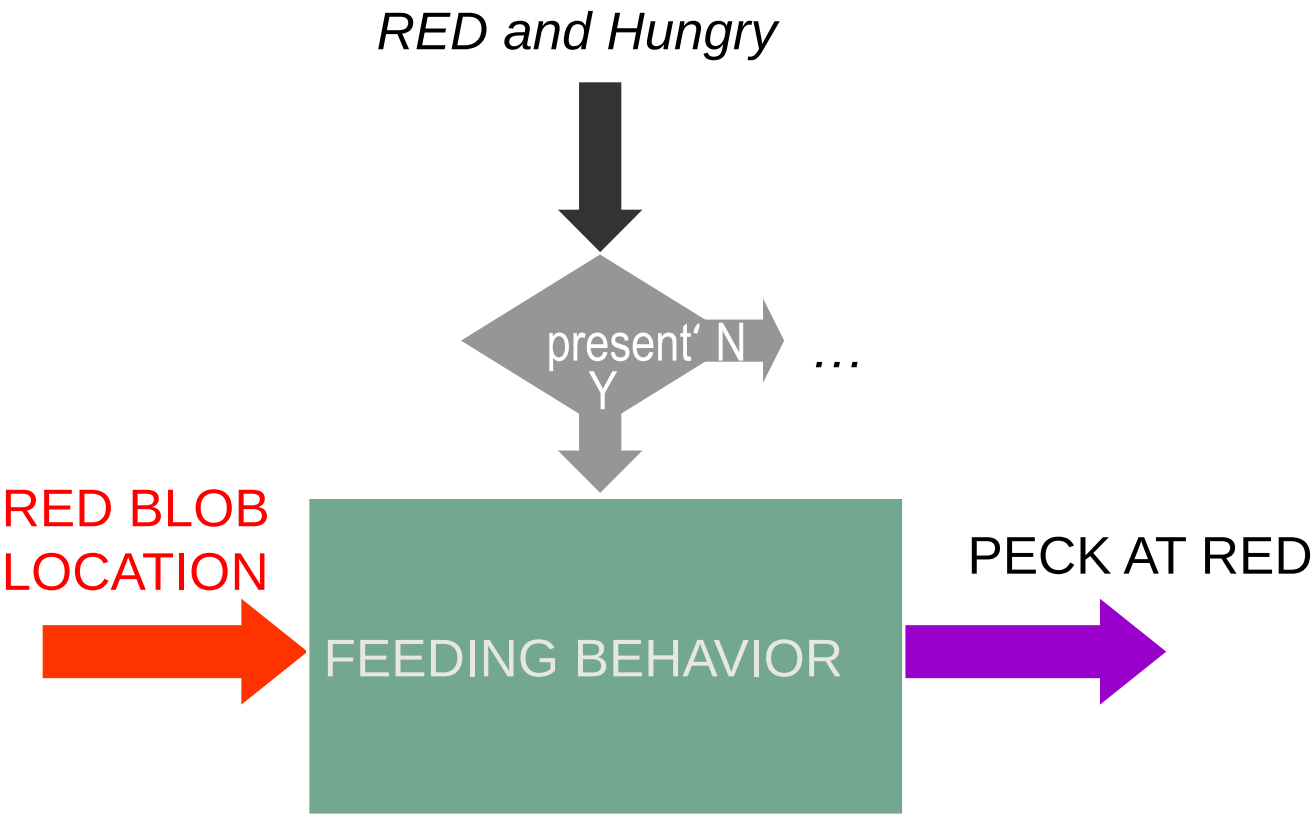
FEEDING BEHAVIOR

PECK AT RED

(Response)

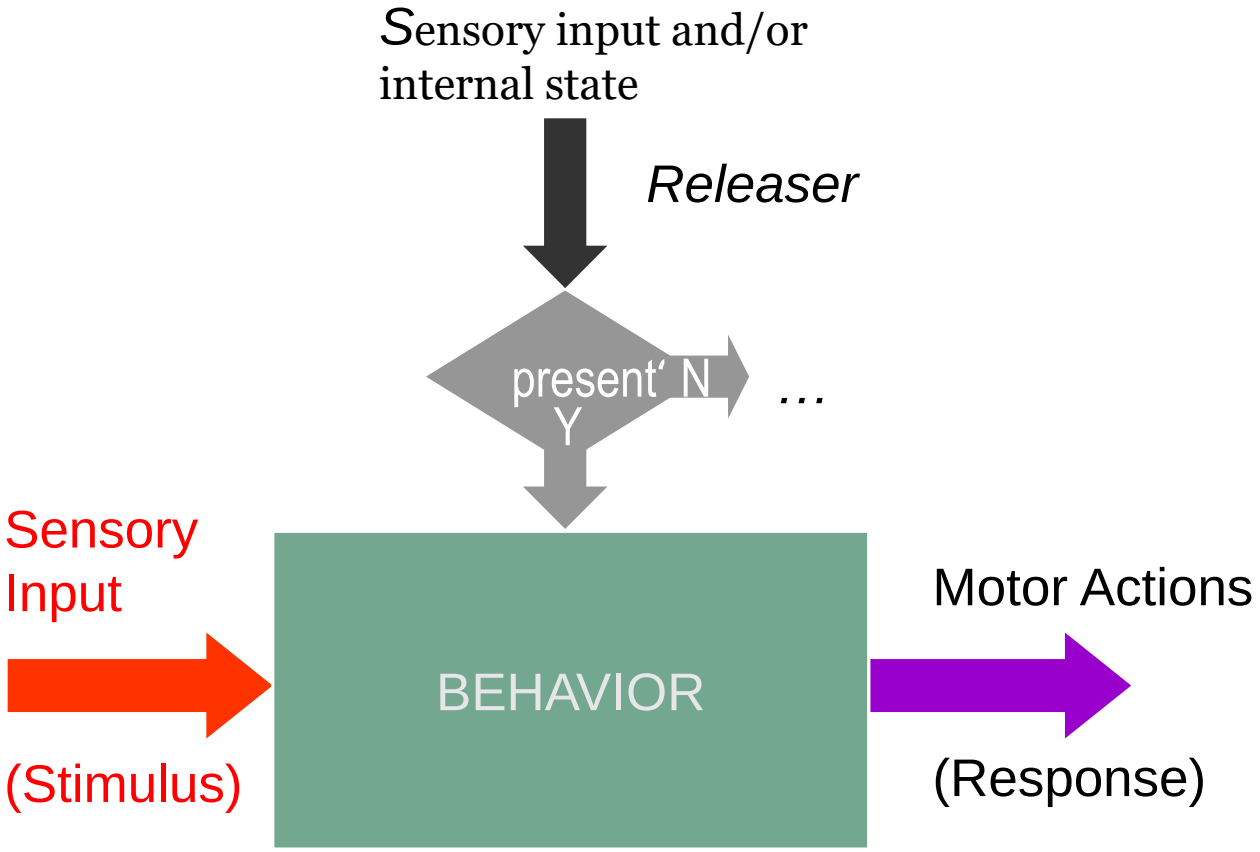
UMEÅ UNIVERSITY

# THE FEEDING RELEASER




UMEÅ UNIVERSITY

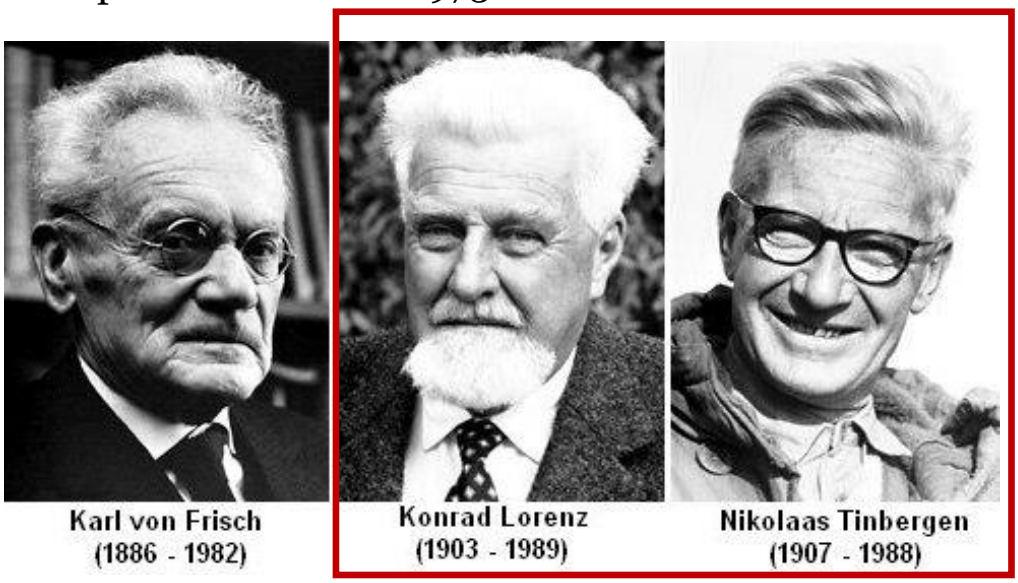
# INNATE RELEASING MECHANISMS (IRM)




UMEÅ UNIVERSITY

# INNATE RELEASING MECHANISMS

Nobel prize in medicine 1973

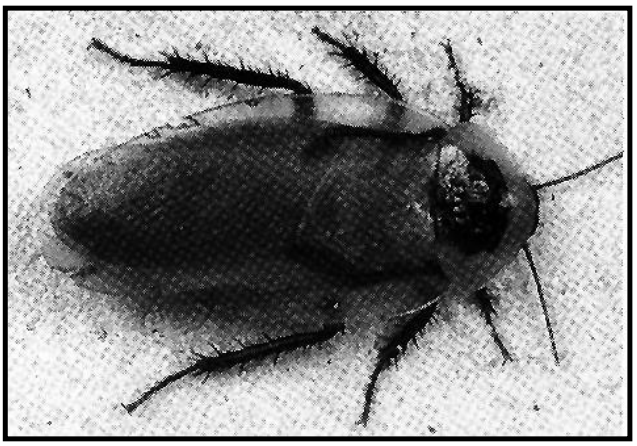



**Founders of ethology**

UMEÅ UNIVERSITY

# EXAMPLE: COCKROACH HIDE

1. Light goes on, the cockroach turns and runs
2. When it gets to a wall, it follows it
3. When it finds a hiding place, goes in and faces outward
4. Waits until not scared, then comes out, even if the lights are turned back off earlier



# RELATED TO WHAT WE'VE LEARNED

1. light goes on, the cockroach turns and runs — *S→R*
2. when it gets to a wall, it follows it — *S→R*
3. when it finds a hiding place, goes in and faces outward — *Taxis*
4. waits until not scared, then comes out, even if the lights are turned back off earlier — *Fixed-action patterns*

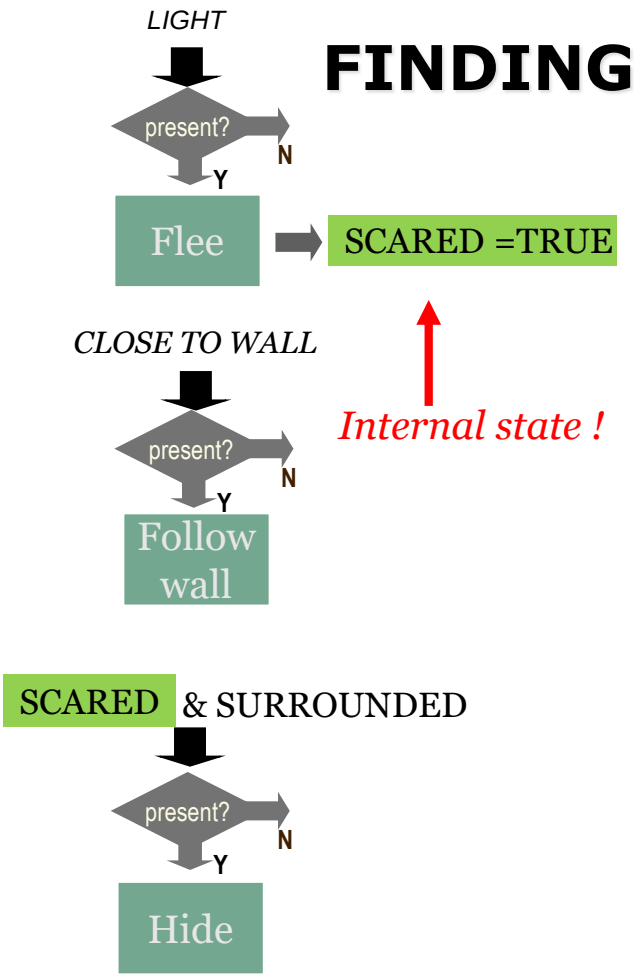
*Sequencing*

# IDENTIFYING BEHAVIORS

| | |
|---|---|
| Flee | light goes on, the robot turns and runs |
| Follow wall | when it gets to a wall, it follows it |
| Hide | when it finds a hiding place goes in and faces outward<br>waits until not scared, then comes out |

UMEÅ UNIVERSITY

# FINDING RELEASERS



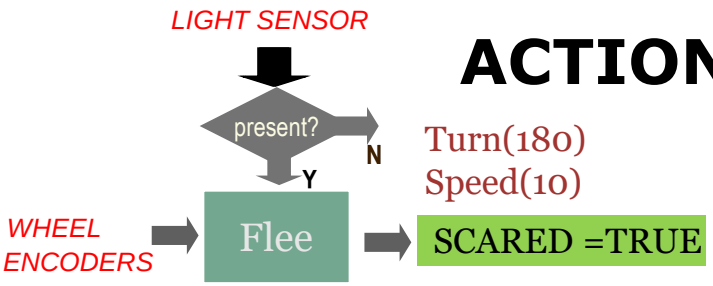

light goes on, the robot turns and runs

when it gets to a wall, it follows it

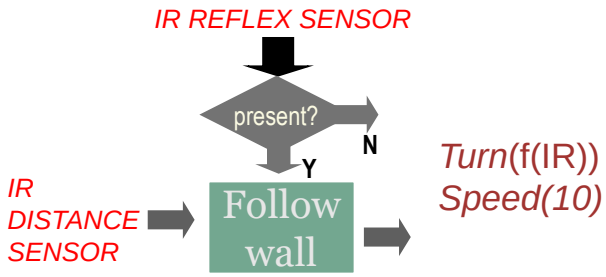
when it finds a hiding place goes in and faces outward.

waits until not SCARED, then comes out

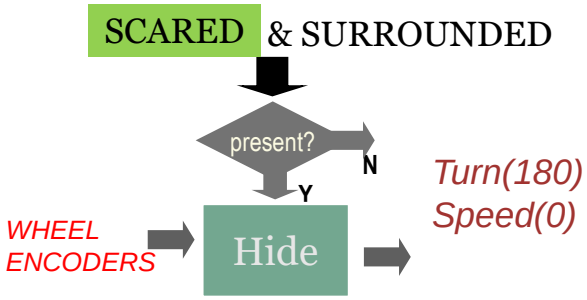
# ACTION AND SENSING




light goes on, the robot turns and runs




when it gets to a wall, it follows it




when it finds a hiding place goes in and faces outward.

waits until not SCARED, then comes out

**Sequencing appears from parallel behaviors activated by releasers**

# NEXT TIME:

**Reactice paradigm**

Potential fields

Subsumption

UMEÅ UNIVERSITY

# LECTURE 13

**Schema theory**

**Reactive paradigm**

**Ola Ringdahl**

UMEÅ UNIVERSITY

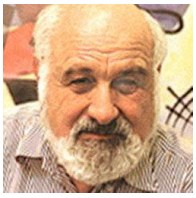
# CONTENTS OF THIS LECTURE

- Biological Foundations of Robot Behavior (part 2)
  - Schema theory
- Reactive paradigm
  - Subsumption
  - Potential fields

# SCHEMA THEORY

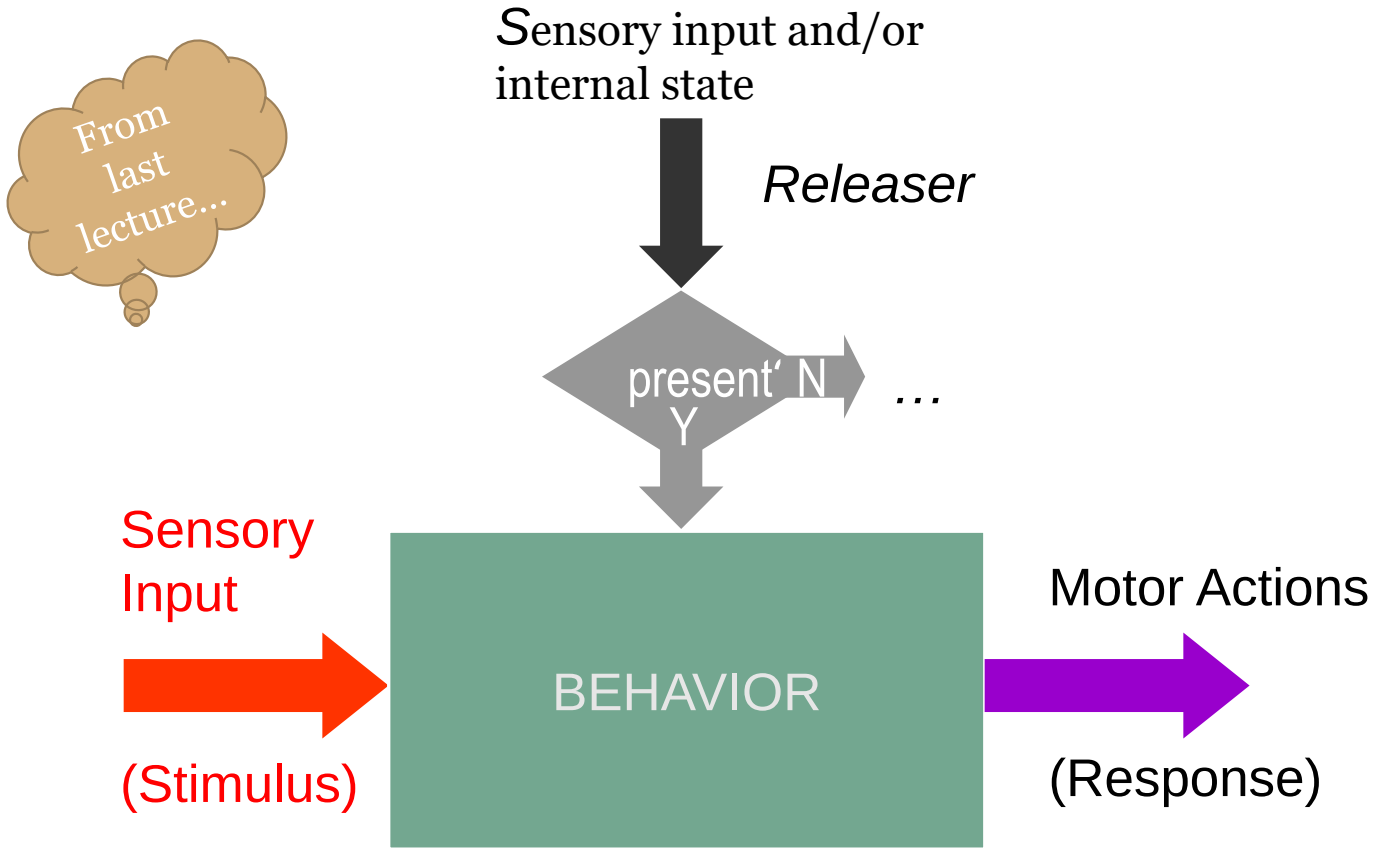
# SCHEMA THEORY

- A way to describe and model behaviours
- Inspired by psychology (used there since the early 1900's)
- Applied to robotics by Michael Arbib



- A schema is a generic template for performing an activity. It contains (compare with OOP):
  - Models and algorithms for how to act and/or perceive
  - Data structures (not in purely reactive behaviours)

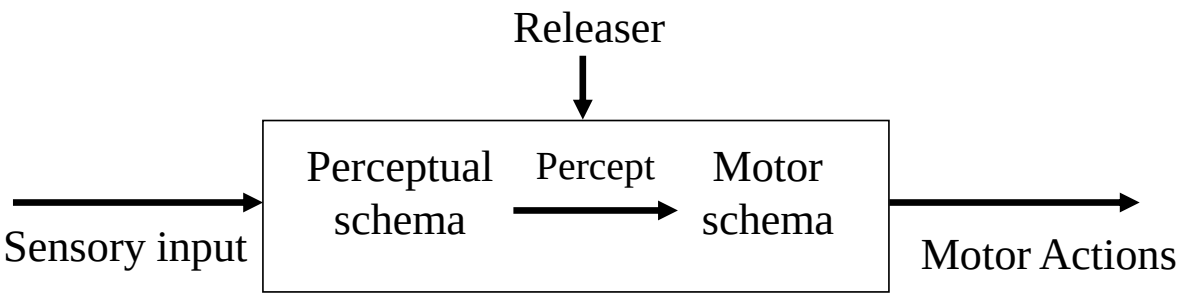
# INNATE RELEASING MECHANISMS (IRM)




UMEÅ UNIVERSITY

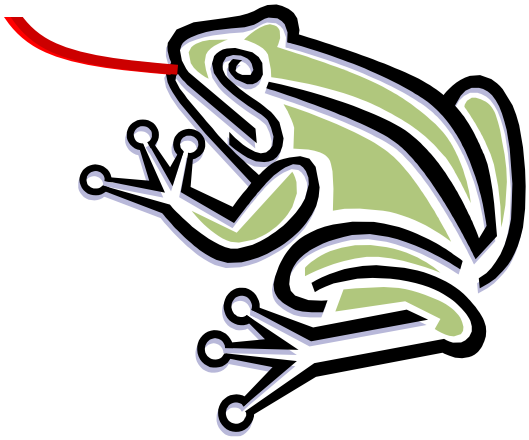
# BEHAVIORS AS SCHEMAS

- The Perceptual Schema produces a Percept
- The Percept is read by the Motor Schema
- Motor actions are vectors: direction and strength



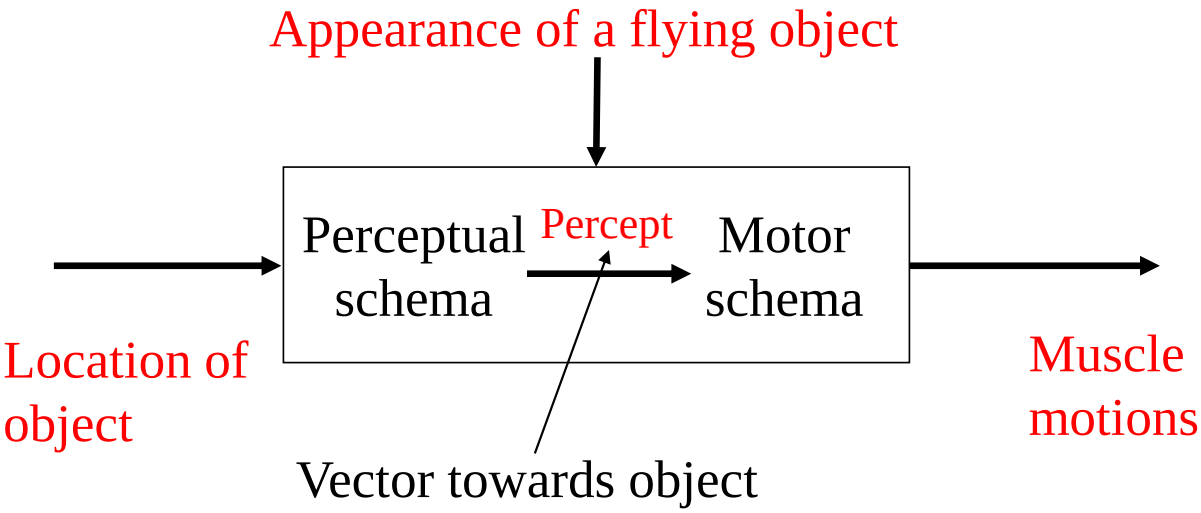

- Several schemas may be active at the same time
- Actions are combined by vector addition

# Fly Snapping Behavior



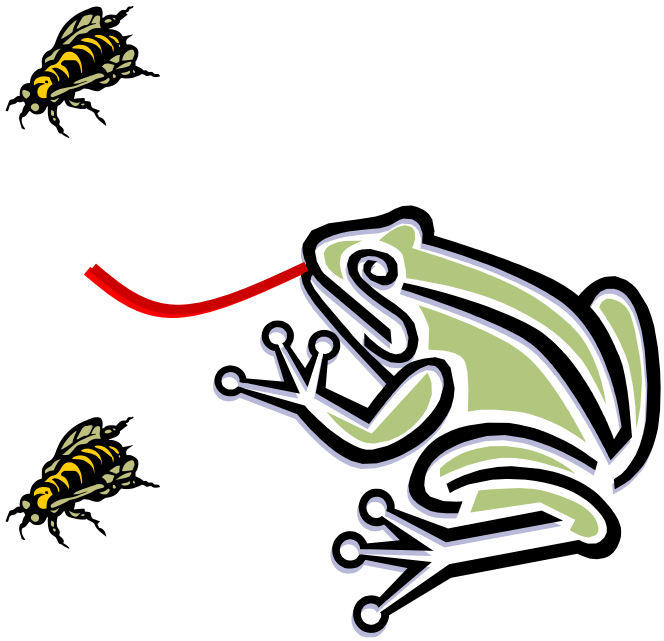
UMEÅ UNIVERSITY

# Fly Snapping Behavior



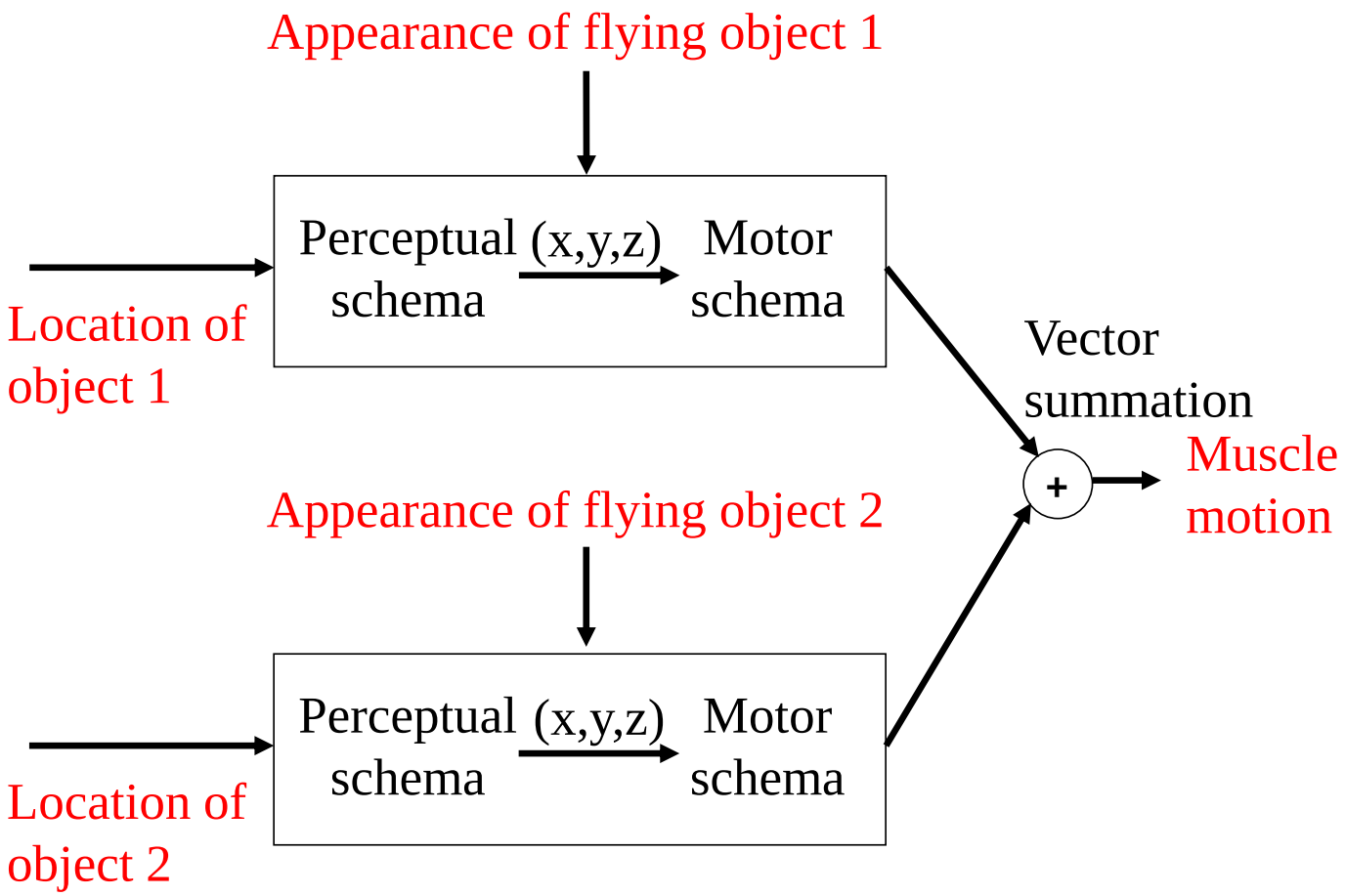

UMEÅ UNIVERSITY

# 2 flies at the same time



UMEÅ UNIVERSITY

# 2 flies at the same time




The toad will indeed snap in the middle!

# 1 fly may be more than enough

# CONCLUSIONS FOR ROBOTICS

- Maybe programming in terms of parallel behaviors is better than STRIPS or other deliberative approaches
- Individual behaviors
  - are often simple
  - are independent
  - can be implemented as threads
- Sequences of behaviors
  - can appear from parallel behaviors activated by releasers (internal states and/or external sensor data)
  - may appear intelligent for an observer
- The system “degrades gracefully” if any part fails since it’s modular
- IRMs and Schemas are nice ways to start thinking about the computational structure of programming a robot

# REACTIVE PARADIGM

## Mataric - Reactive paradigm (p163-169)

UMEÅ UNIVERSITY

# THE REACTIVE PARADIGM

- Reacted against classical AI
  (the hierarchical paradigm/ deliberative systems)
- Reduced emphasis on:
  - Knowledge representation
  - Planning
- Increased emphasis on:
  - Sensing and acting within the environment
  - ”Situatedness” - interaction with the environment is crucial for intelligent behavior
  - ”Embodiment” – the structure of the body is tightly connected to behavior and intelligence.

# THE REACTIVE PARADIGM

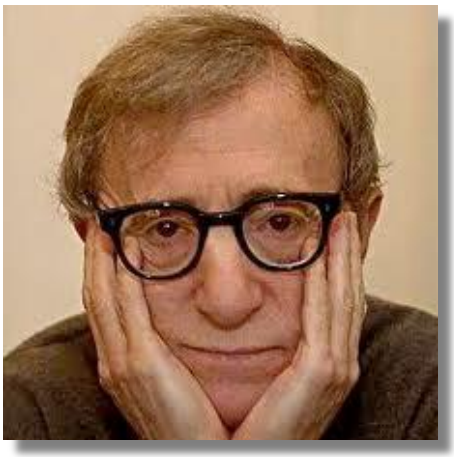
**Suggested by Rodney Brooks MIT (1986)**

- "Planning is just a way of avoiding figuring out what to do next"
- "The world is its own model"
- "Elephants don't play chess"
- Complex control systems are not necessary for complex behavior
- Intelligence is in the eye of the observer
- No representation. No calibration. No complex computers. No High bandwidth communication

# RELATED QUOTES

*”Everybody’s got a plan until they get hit”*

Mike Tyson – unknowingly commenting on *The closed world assumption*



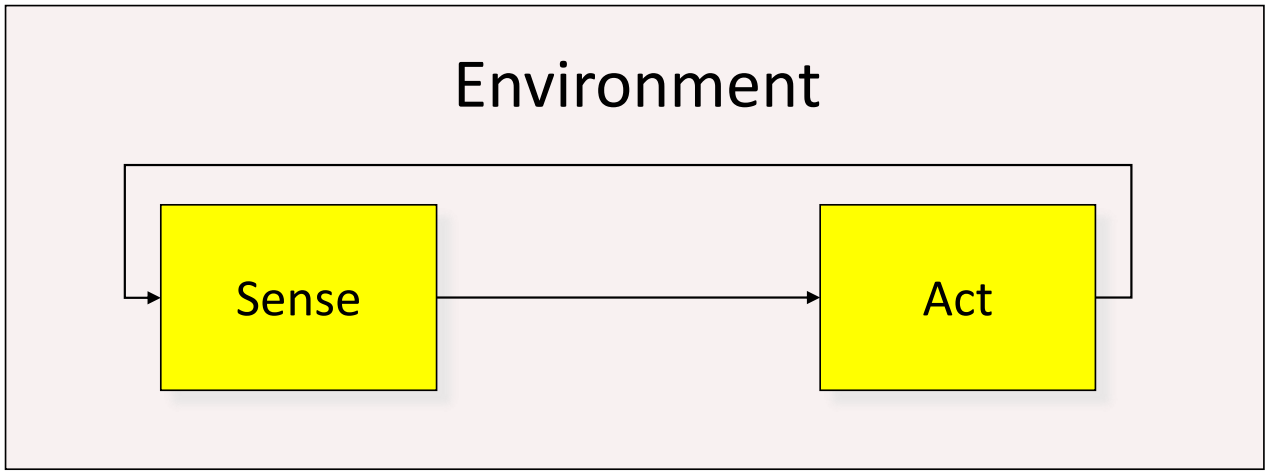
*"If you want to make God laugh, tell him about your plans”*

Woody Allen – general disbelief in predictions of the future

UMEÅ UNIVERSITY

# THE REACTIVE PARADIGM

**Walter 1953, Breitenberg 1984, Brooks 1986**




Method: "Sense-Act

- Sensor data is immediately transformed into Actions
- Based on biology (e.g. insects)
  - Stimuli → Response

# THE REACTIVE PARADIGM

- The basic components are called *BEHAVIORS*
- Each behavior
  - is implemented as reactive rules, mapping stimuli to respones without thinking or computing
  - has no (or little) memory
  - computes responses continously
- All behaviors exist in parallell and are activated (triggered) by percepts from the sensors
- *Coordination* if several behaviors are active at the same time

# ARCHITECTURAL ISSUES

- How to represent behaviors
- How to trigger behaviors
- How to coordinate behaviors

<u>Two common architectures:</u>

- Subsumption architecture
- Potential fields

# REACTIVE PARADIGM ALGORITHMS

## Subsumption

## Potential fields

UMEÅ UNIVERSITY

# Subsumption Architecture

## Suggested by Rodney Brooks MIT (1986)

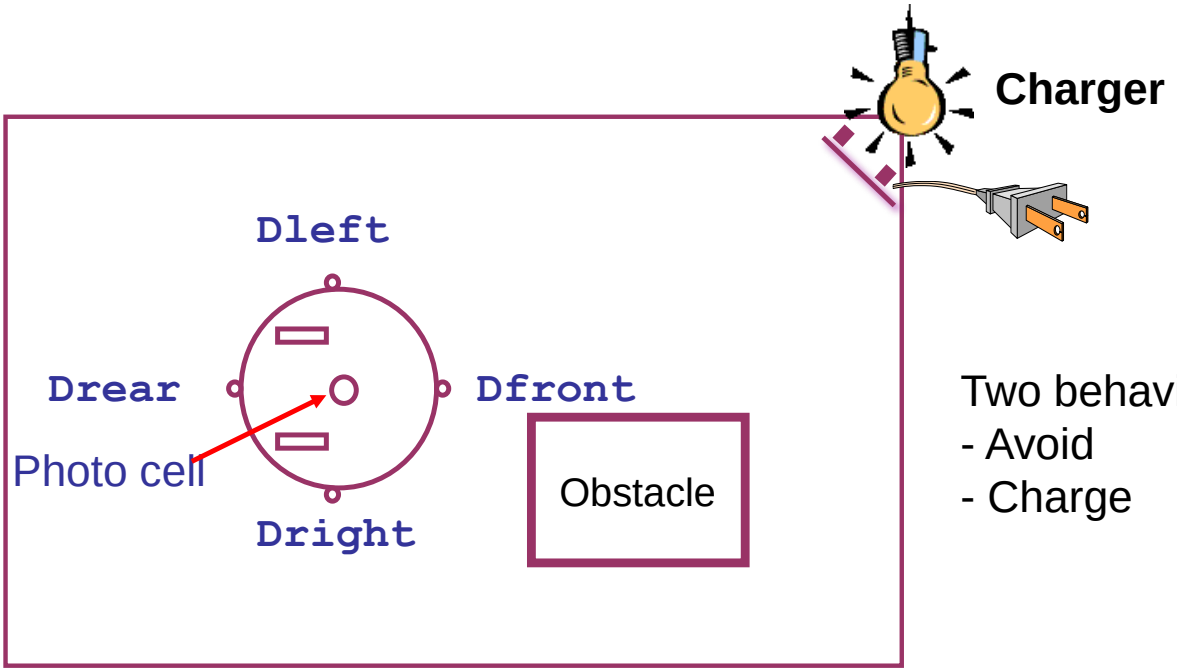
UMEÅ UNIVERSITY

Example

# Subsumption Architecture

**A self-charging robot**

- 2 engines for propulsion and stearing
- Controlled by speed and angular velocity
- Battery voltage sensors `Bat`
- 4 ultrasonic distance sensors `Dleft,Dright,Dfront,Drear`
- 1 photocell that gives the angle `D` to the light source




Two behaviors:
- Avoid
- Charge

Example

# Subsumption Architecture

## Avoid

US sensor → *Avoid* → Motors

The behavior *Avoid* contains code that maps the signals from the 4 US sensors to the actuator (Motors):

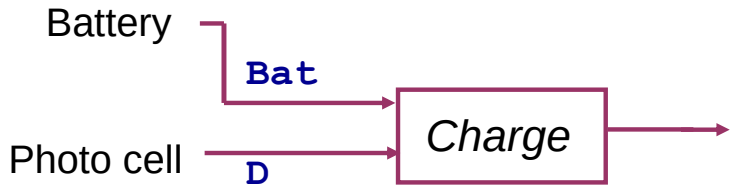
```
If Dfront<d0 or Drear<d0 or Dleft<d0 or Dright<d0
   % Act only if an obstacle is close
   if Drear<min(Dleft, Dright, Dfront) and Dfront>S0
     "move ahead"   % Go away from obstacles
   else
      if Dleft<Dright then "turn right"
      if Dright<Dleft then "turn left"
   end
end
```

Note: responses are computed continously (don't wait until finished)

Example

# Subsumption Architecture

## Charge




The behavior *Charge* contains code that turn the robot towards a light source (the charger), move forward and dock.

The PhotoCell senses light in direction **D** (**D** = 0 if light is straight forward)

For instance:

```
If Bat<b0 and
   if D>Dtol then "turn right"
   if D>-Dtol then "turn left"
   "move ahead"
End
```

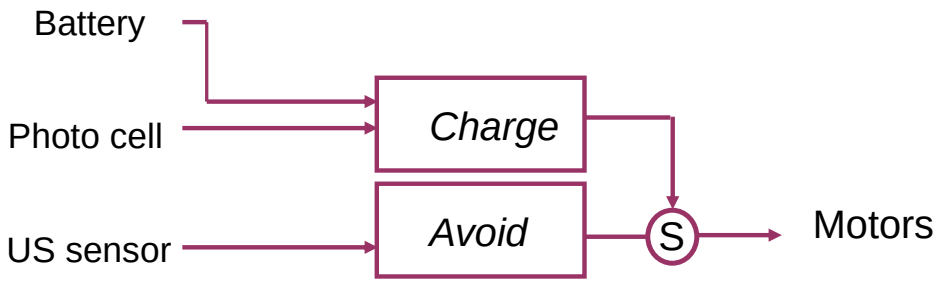
Note: responses are computed continously (don't wait until finished)

UMEÅ UNIVERSITY

Example

# Subsumption Architecture

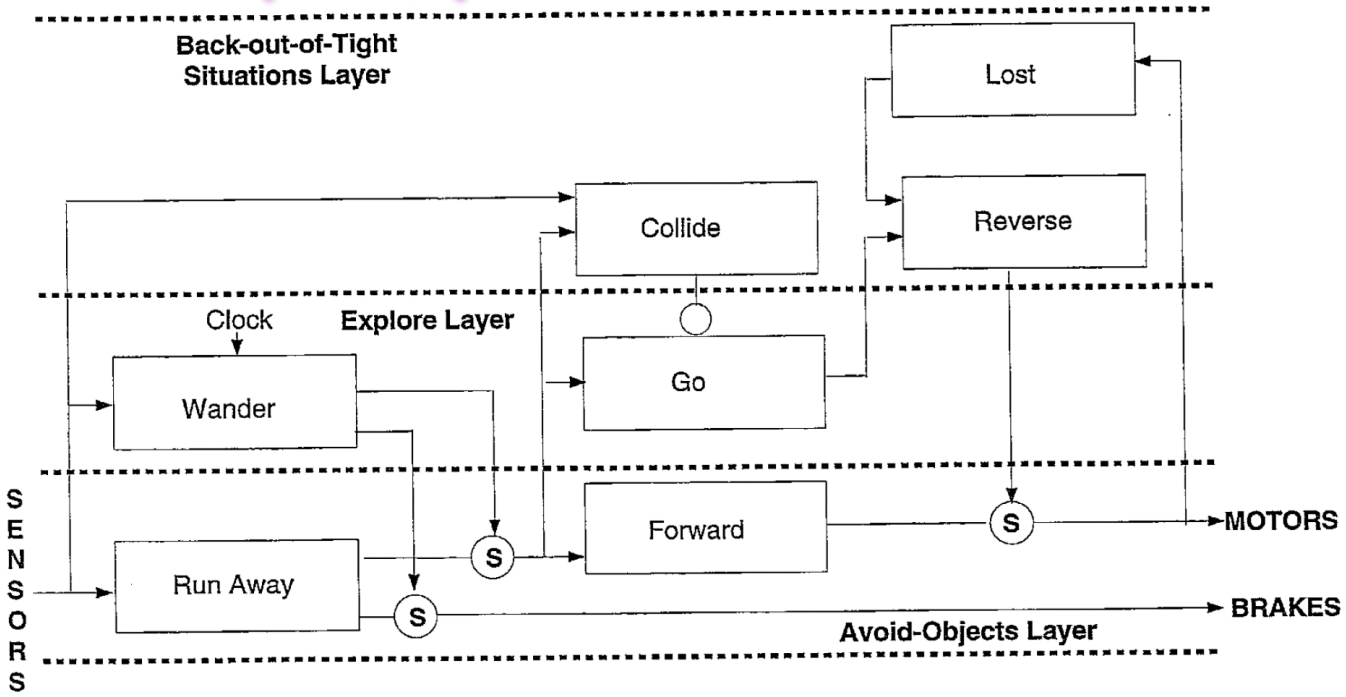
## Coordination of Behaviors




- S is a *suppressor node* that blocks and replaces messages from the *Avoid* behavior iff *Charge* sends messages at the same time.
- *Charge* subsumes (contains/*innesluter*) *Avoid*
- *Avoid* is subsumed by *Charge*

# Subsumption Architecture

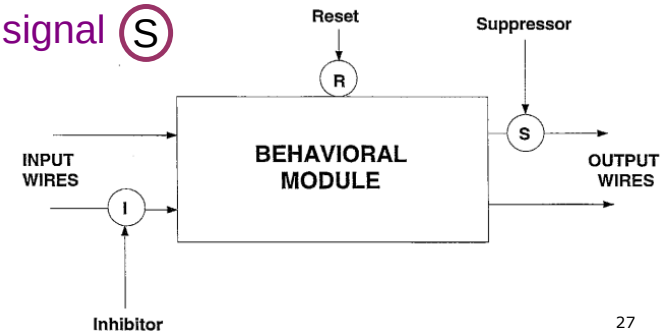
## A more complex example




Each layer has a separate goal

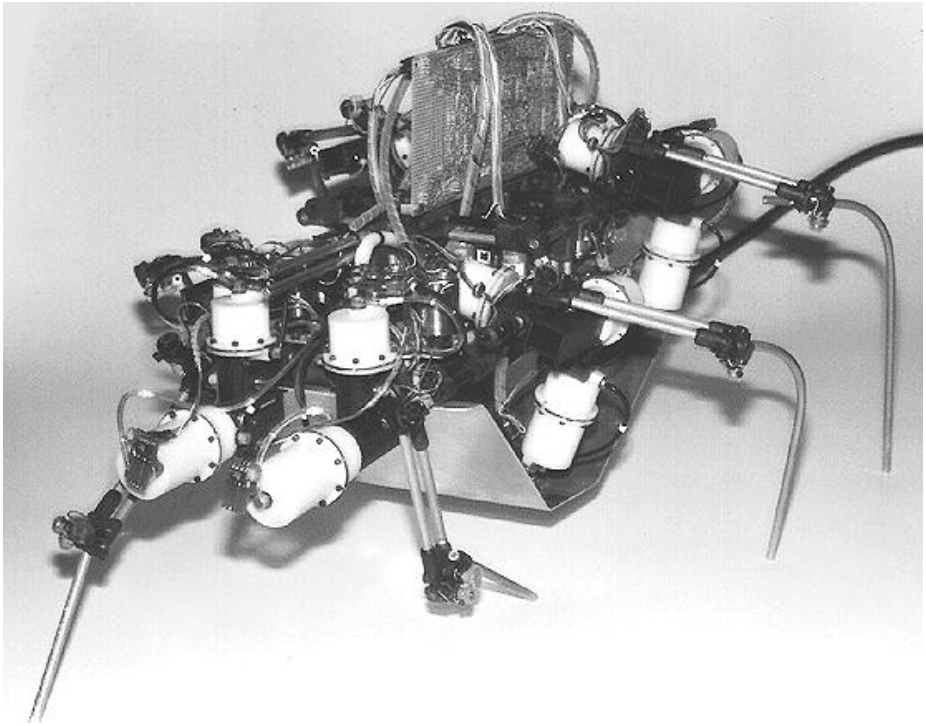
# Subsumption Architecture

- Each task-achieving layer contains one or several reactive behaviors
- (almost) no internal states
- Loosely coupled layers:
  - The lower layers have no awareness of higher levels (i.e.: bottom-up design)
  - Higher layers may access lower levels, but very low bandwidth. “The world serves as primary medium of communication”
- Competitive coordination with two mechanisms:
  - Inhibition. Prevents a signal from being transmitted I
  - Suppression. Blocks and replaces a signal S

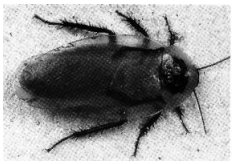



UMEÅ UNIVERSITET

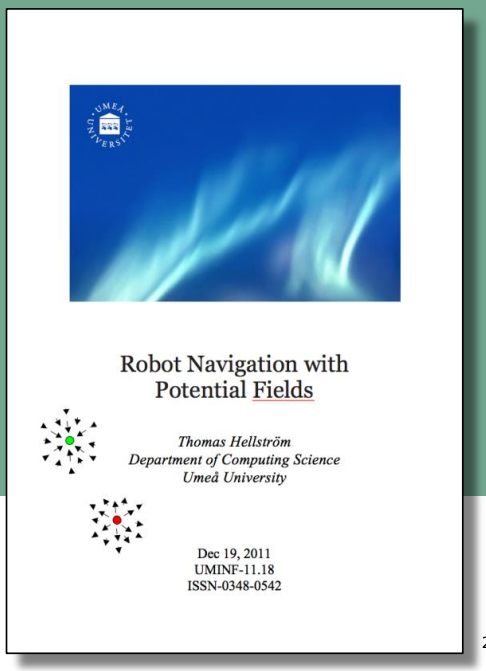
# A robot cockroach



- A six-legged, pneumatically powered walking robot based on the Subsumption architecture with 57 behaviors
- The robot's mechanics are modeled after a cockroach, Blaberus Discoidalis.



# POTENTIAL FIELDS FOR ROBOT NAVIGATION


Robot Navigation with Potential Fields

Thomas Hellström
Department of Computing Science
Umeå University

Dec 19, 2011
UMINF-11.18
ISSN-0348-0542

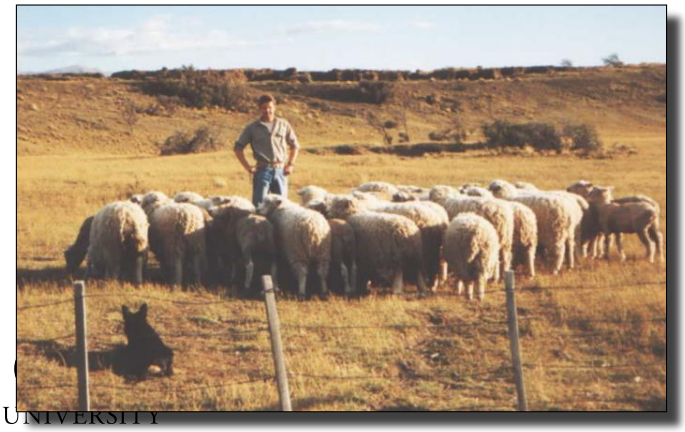
UMEÅ UNIVERSITY

# POTENTIAL FIELDS

- A reactive technique for guiding a robot from A to B
  - No Planning!
- Inspired by biology
- Responses are represented in a uniform format:
  Vectors with Strength (force or velocity) and Orientation
- Can be used to implement Motor schemas

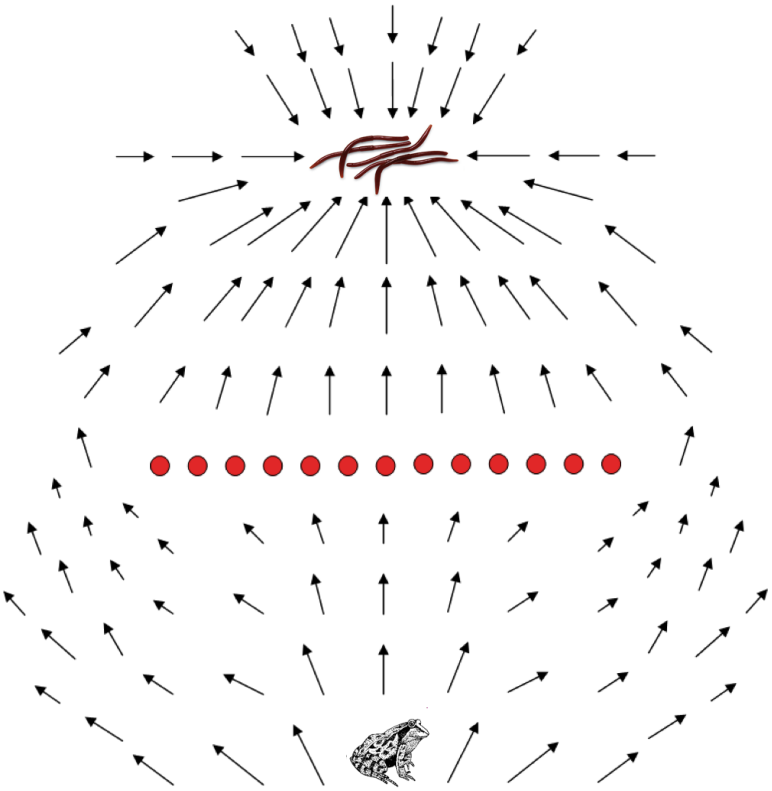
# INSPIRED BY BIOLOGY

Describe the “forces” emanating from the  shark and men below



# INSPIRED BY BIOLOGY

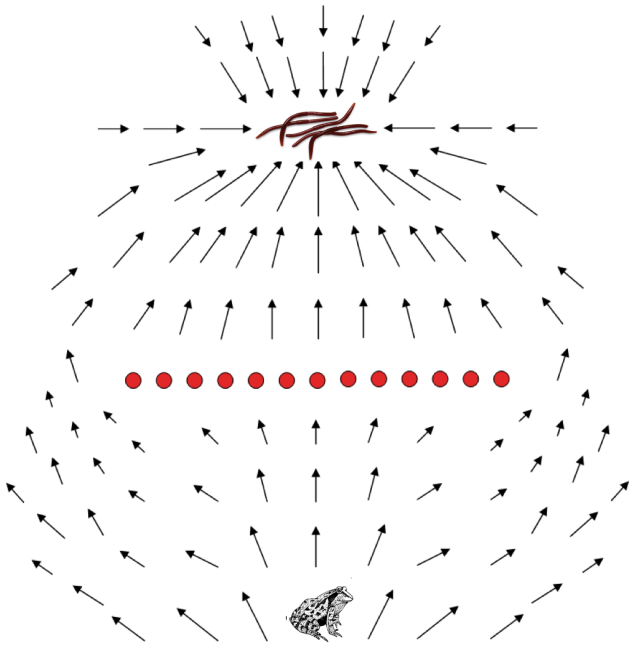
- Motion of toads attracted to a can of worms
- Model: Attraction and repulsion



UMEÅ UNIVERSITY

# THEORY

- A robot should navigate over a 2D field occupied by an obstacle, until it reaches a goal position G.
- For each point (x,y) we define two values given by *potential field functions*, e.g.:
  - $U_{rep}$: 1/(distance to obstacle)
  - $U_{att}$: distance to G
- We want to move in the direction of decreasing $U_{rep}$ and $U_{att}$



UMEÅ UNIVERSITY

# THEORY

- For a function f, the gradient vector $\nabla$f points in the direction of increasing f.
- We define *force vectors F* as the negative gradients of $U_{rep}$ and $U_{att}$:

$$F_{rep} = -\nabla U_{rep} = -\left( \frac{\partial U_{rep}}{\partial x}, \frac{\partial U_{rep}}{\partial y} \right)$$

$$F_{att} = -\nabla U_{att} = -\left( \frac{\partial U_{att}}{\partial x}, \frac{\partial U_{att}}{\partial y} \right)$$

- $F_{rep}$ points in the directions of smaller $U_{rep}$ (away from the obstacle)
- $F_{att}$ points in the directions of smaller $U_{att}$ (towards G)
- A strategy to move safely towards G is to move in the direction $F = F_{rep} + F_{att}$

# THEORY

<u>Generalization:</u>

- More obstacles
- Other types of potential fields

The general expression for **F** computed at a point (x,y) is:

$$\mathbf{F}(x, y) = \sum_i \mathbf{F}_i(x, y)$$

where $\mathbf{F}_i(x,y)$ is the force vector associated with potential field $U_i$, i.e. the negative gradient of $U_i$ : $\mathbf{F}_i(x,y) = -\nabla U_i(x,y)$

# AN ALGORITM FOR NAVIGATION

<u>An algorithm for robot navigation from a position ($x$,$y$):</u>

1. Represent the goal and all obstacles as potential fields.
2. Compute force vectors $\mathbf{F}_i(x,y)$ for all potential fields
3. Sum all force vectors into $\mathbf{F}(x,y)$
4. Move along $\mathbf{F}(x,y)$ with speed proportional to $|\mathbf{F}(x,y)|$
5. Repeat from 2. until goal is reached

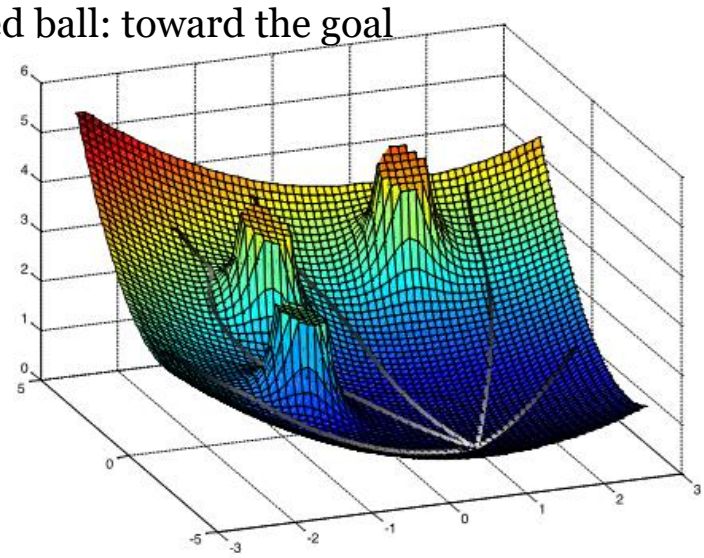
IMPORTANT:

- Normally each $\mathbf{F}_i$ is computed as a function of distance and direction to objects around the robot (from sensors)
- $\mathbf{F}_i$ only has to be computed for current position (x,y)

# PHYSICS ANALOGY

A ball finding its way to the bottom of a valley filled with obstacles

- One attractive goal at (1, -2) at the bottom of the valley
- Three repulsive objects as “steep rocks”
- The paths show the motion of the imagined ball: toward the goal avoiding the obstacles

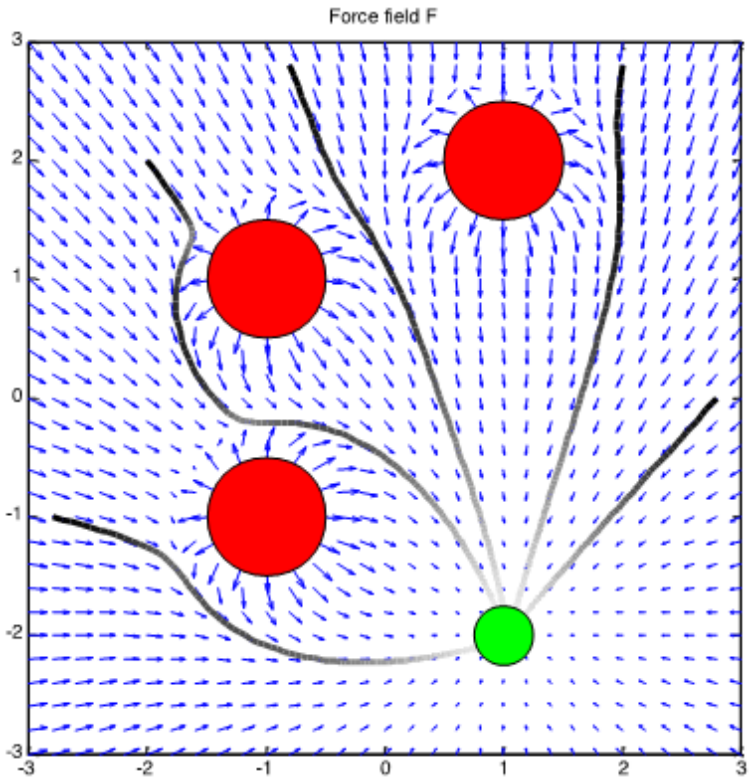



The paths also corresponds to the potential field theory:

- z-axis is the sum $U$ of all potential fields
- Direction of maximum descent ($-\nabla U$)

# PHYSICS ANALOGY CONT.

- $\mathbf{F} = -\nabla U$ is the force that causes the ball to move
- Each arrow represents the direction and magnitude of **F** at the location of the arrow
- The robot follows the direction of **F** in every point
- Speed is given by $|\mathbf{F}|$, the magnitude of **F** (shown as gray scale of paths)
- Compare with the toad and worms

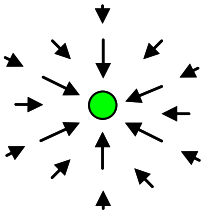



Red: repulsive obstacles

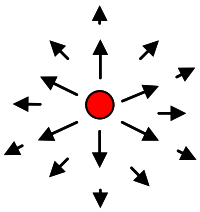
Green: attractive goal

# BASIC TYPES OF POTENTIAL FIELDS

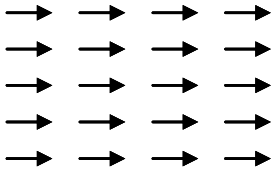
Attraction



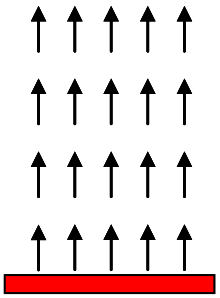
Repulsion



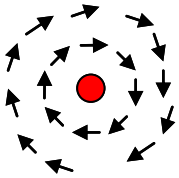
Uniform



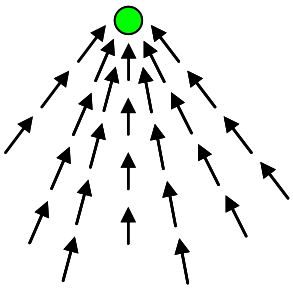
Perpendicular



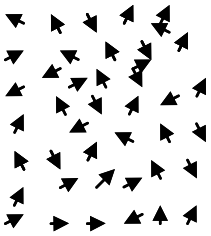
Tangential



Selective attraction



Random



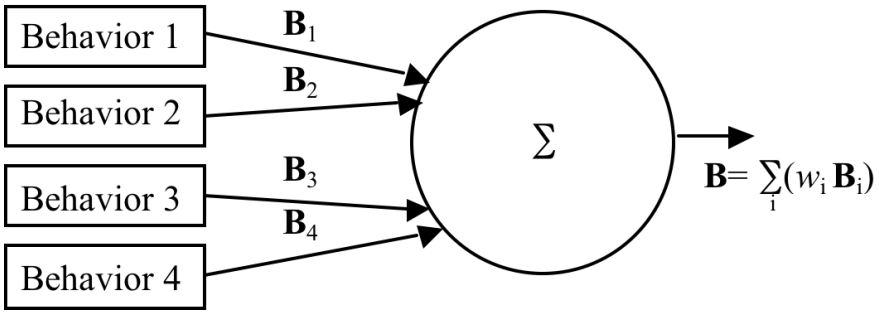
- The figures show the force field **F**, i.e. the gradient of the potential field function
- Each arrow indicates strength and direction of the force
- The force fields are often, somewhat sloppily, denoted potential fields

# FIELDS AS BEHAVIORS

Basic idea: Follow the force!

- One field can be thought of as one behavior
- Vector summation to create compound behaviors

# FIELDS AS BEHAVIORS

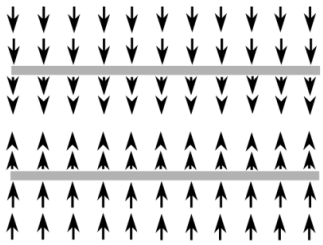



- Four behaviors; each one represented by a potential field, or recursively by other coordinated behaviors
- The output force vectors $\mathbf{B}_1,\ldots,\mathbf{B}_4$ are summed and weighted to produce one compound vector output $\mathbf{B}$
- Note: $\mathbf{B}$ needs to be computed for the robot's location only

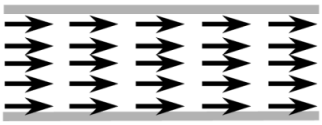
# EXAMPLE: FOLLOW-CORRIDOR

**constructed as a sum of 3 potential fields:**

2 x Perpendicular + Uniform

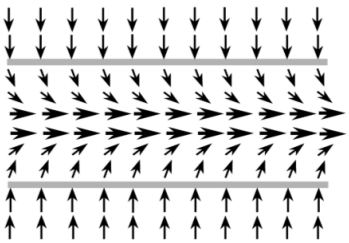


a



b

=



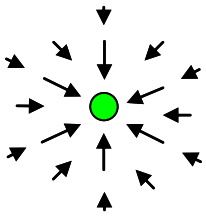
c

The robot “feels” the vector for the current location, then moves accordingly, “feels” the next vector, moves, etc.

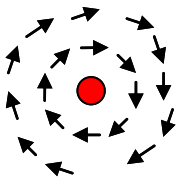
# EXAMPLE: DOCKING BEHAVIOR

- Constructed as a sum of 3 potential fields:

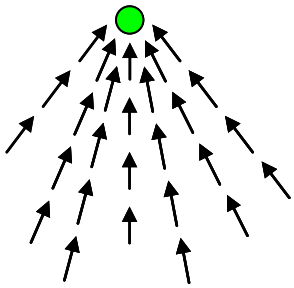
Attraction

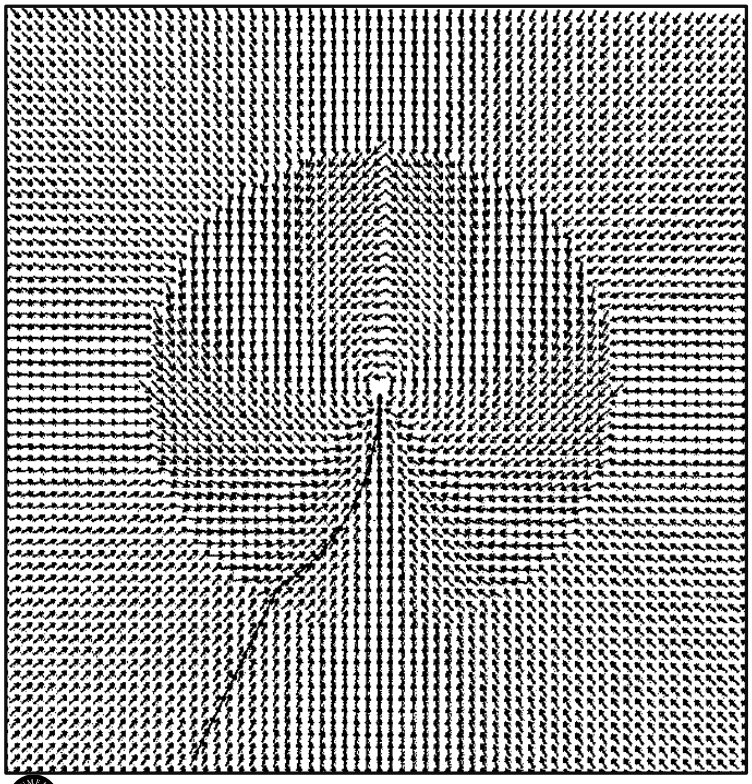


Tangential



Selective attraction



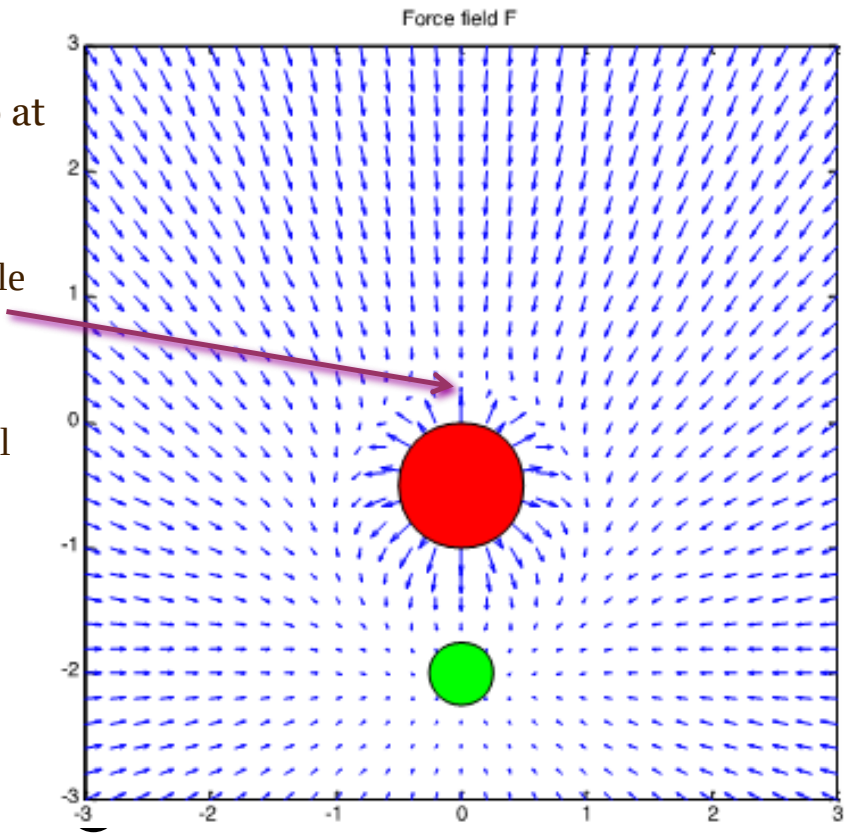


# PROBLEM: LOCAL MINIMA

Sometimes the fields add to zero at some points!

<u>Example:</u>
The repulsive force from the obstacle cancels the attractive force from the goal.

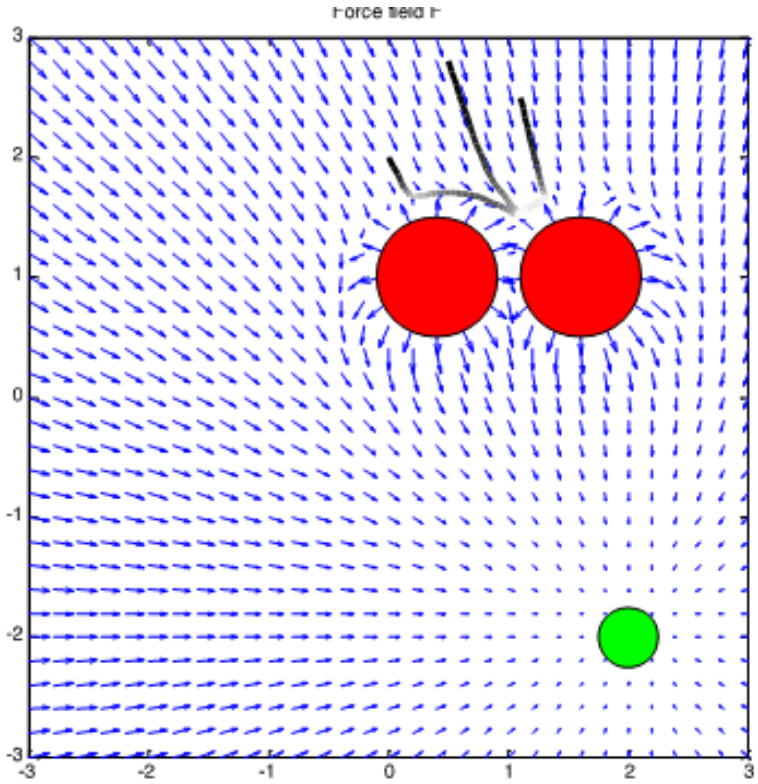
If the robot reaches this point, it will stop!



# PROBLEM: LOCAL MINIMA

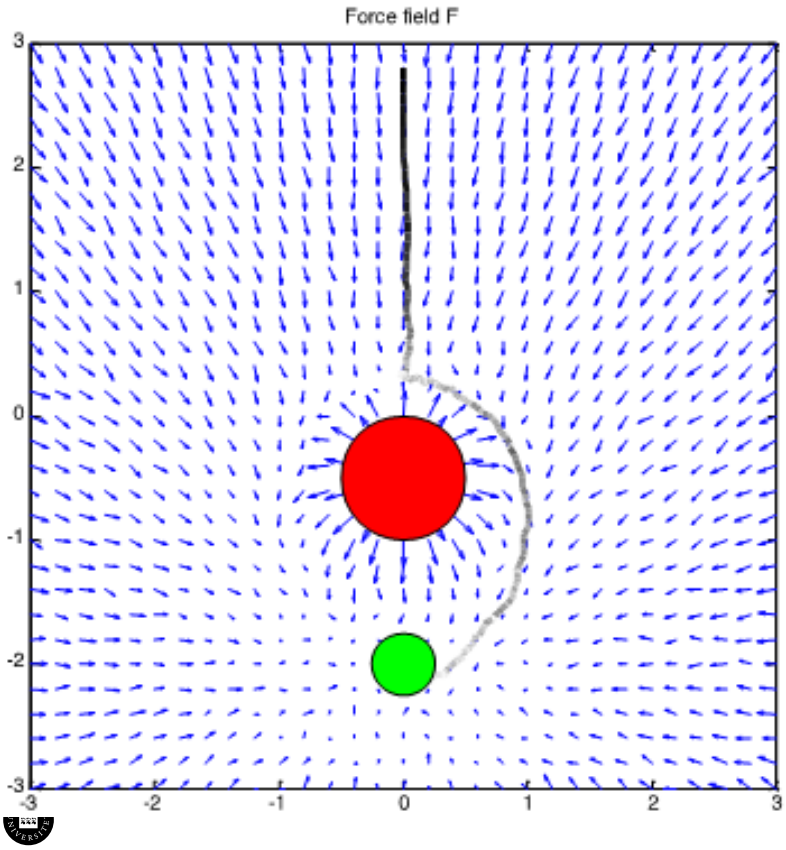
- The robot may be dragged into a local minimum from several different starting positions.

- The gray scale of robot paths corresponds to the speed of the robot.



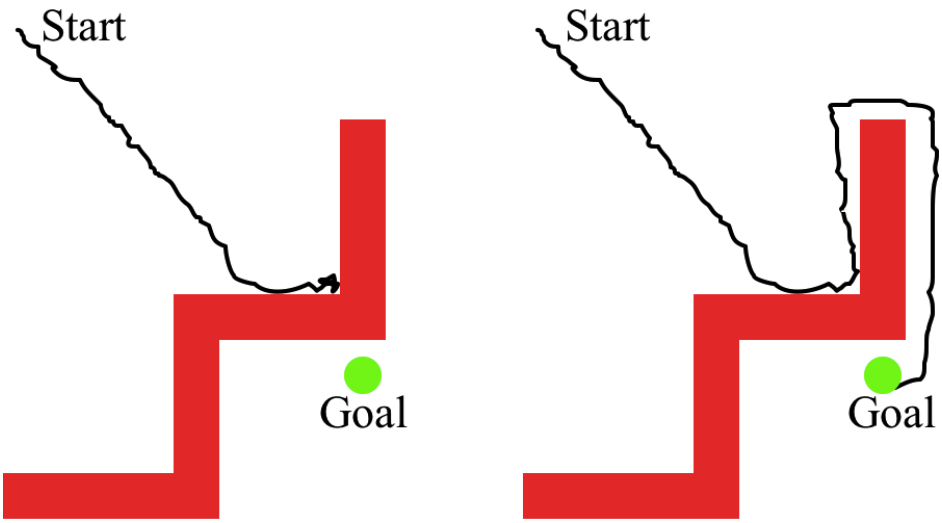
# SOLUTION 1: LOCAL MINIMA

- The local minimum is avoided by <u>adding a potential field comprising random values.</u>
- The jagged path is a result of the random force vectors.



# SOLUTION 2: LOCAL MINIMA

- Sometimes, adding noise will not solve the problem but rather introduce a **cyclic behavior**: the robot keeps returning to the local minimum
- Solution: Adding a dynamically updated potential field with <u>repulsive forces from all previously visited locations</u>

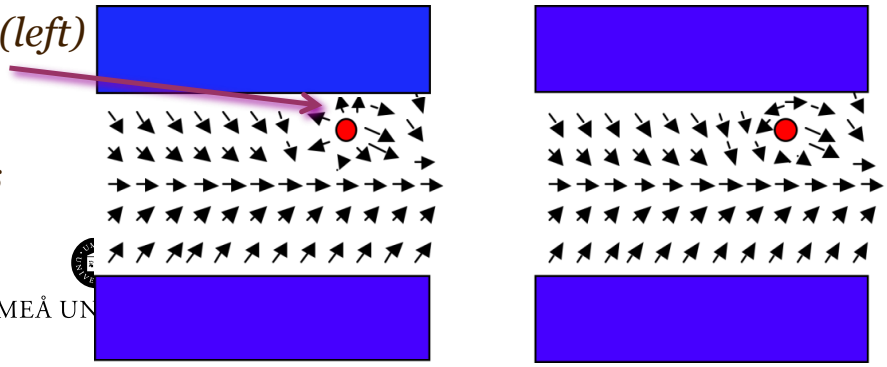


# SOLUTION 3: LOCAL MINIMA

- The regular repulsive fields are replaced by a special kind of tangential fields : *Navigation templates*
- The direction of the tangential field is set in real-time not to counter the vector sum of all other fields
- Reduces the risk of attractive and repulsive forces cancelling each other
- Another benefit: Obstacles will be avoided by a force vector that points in the same direction as the general behavior:

*The regular repulsive potential field (left) may push the robot into the water*

*A navigation template (right) avoids this by forcing the robot to avoid the obstacle in a safe direction*



# PROBLEMS WITH POTENTIAL FIELDS

- Jerky motion (depends on update rates)
- Robot and obstacles treated as points (=> risk of collission)
- The robot is expected to change velocity and direction instantaneously
- Sensitive to local minima

  Three approaches:
  - Injecting randomness
  - Adding an “avoid-past” schema
  - Navigation templates
- Cyclic behavior

  One approach:
  - Adding an “avoid-past” schema that generates a repulsive force from recently visited areas

# ADVANTAGES WITH POTENTIAL FIELDS

- Easy to implement and visualize
  - Robot behavior easy to predict by designer
- Support for parallelism
  - Each field is independent of the others and may be implemented as general software, or even hardware, modules
- They can be easily parameterized and configured, also during design or in real-time
- The combination mechanism is flexible and can be tweaked with gains to reflect varying importance of sub-behaviors

# COORDINATION OF BEHAVIORS

**A general problem in reactive systems is how to handle concurrent and disagreeing behaviors**

# COORDINATION OF BEHAVIORS

**Two major types of coordination mechanisms:**

**Competetive**. “Winner takes all”, “Arbitration”

- Fixed priority methods
  - Subsumption
  - ”Subsumption light”
- Dynamic priority methods
  - Activity based; Select the most “active” behavior.
  - Voting based; each behavior suggests MANY responses

**Cooperative**. “Behavioral fusion”

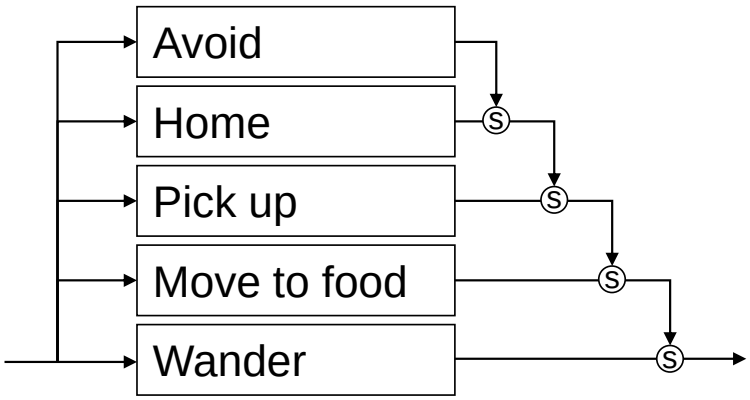
- The representation of behaviors must allow fusion.
  - Vector addition such as in potential-fields and schema based approaches
  - Vector addition with dynamic gains
  - Fuzzy logic

# "SUBSUMPTION LIGHT"

## Competitive - Fixed priority

**Foraging example:**

- *Wander*: always active
- *Move to food*: active if the robot sees the food
- *Pick up*: active if food is within reach
- *Home*: active if the robot has food
- *Avoid*: active when an obstacle is in front of the robot





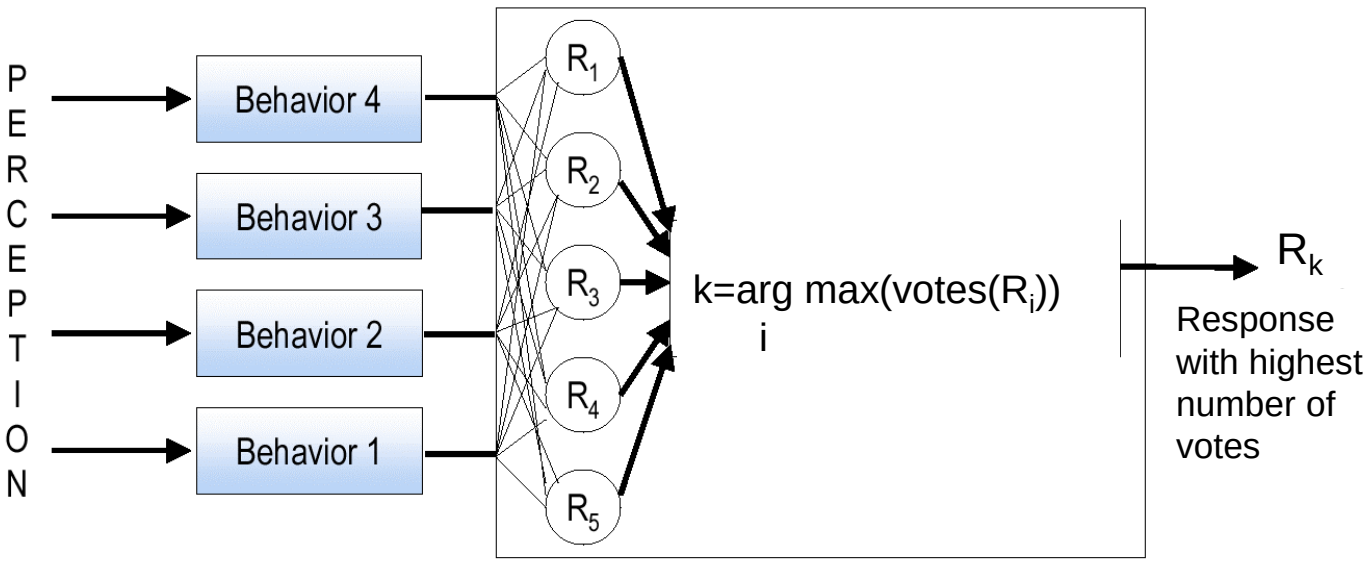

Example from: Srividhya Rajendran, The University of Texas

# Voting based arbitration

## Competitive - Dynamic priority

- One pre-defined set of discrete Responses for ALL behaviors
- Each behavior votes for each response
- The response with most votes is executed

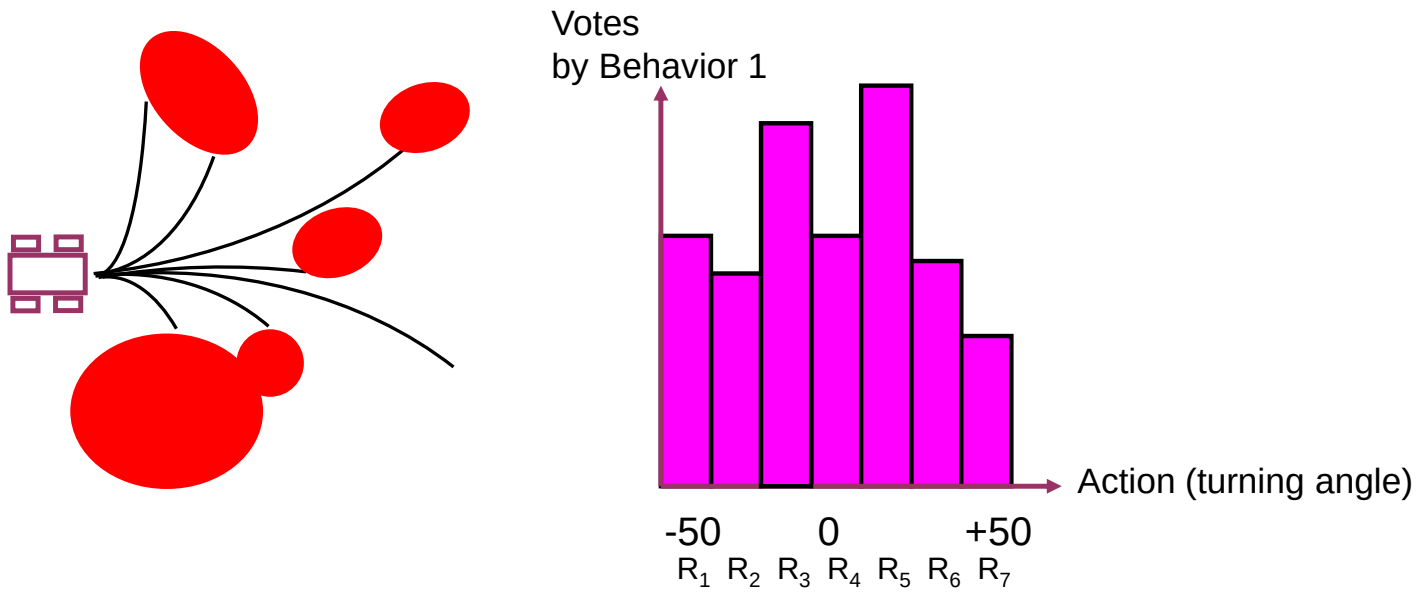
Priorities (no. of votes) may vary during mission




Voting-based coordination

# EXAMPLE VOTING

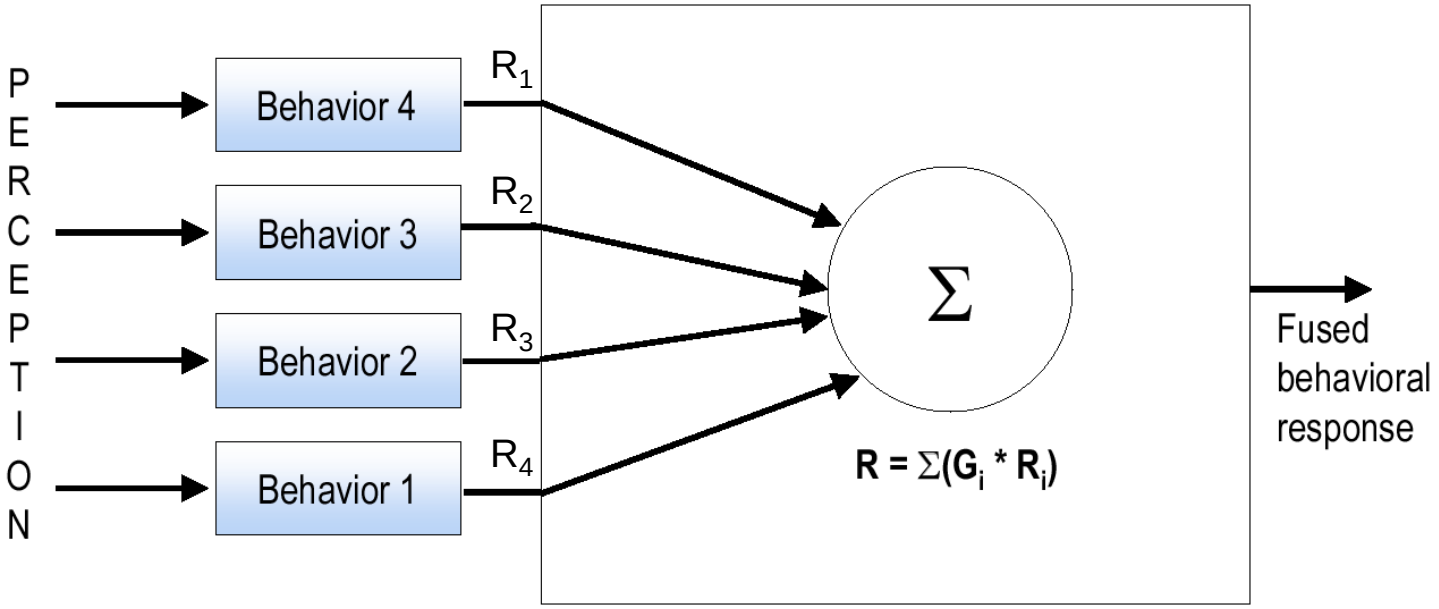
- DAMN – A Distributed Architecture for Mobile Navigation
- Behavior 1 does obstacle avoidance and votes for each response based on estimated time to collision in that direction.
- Behavior 2 does homing and votes for each response based on direction to goal
- Responses R1 to R7 : turning angle



# Vector Addition

## Cooperative – dynamic gains

The behaviors are fused using gains $G_i$. Gain levels may vary during mission and depend on the robot's goals and percepts




Behavioral fusion

# THE REACTIVE PARADIGM

Advantages:

- Fast reaction time
- Requires less memory and CPU power
- Easy to implement and expand
  - Behaviors are independent
  - No modelling necessary
- Makes no Closed World Assumption
  - Works in dynamic environments that are difficult to characterize and contain a lot of uncertainty

# THE REACTIVE PARADIGM

Disadvantages:

- Interacting behaviors can be unpredictable
  *"fast, cheap and out of control…"* (Rodney Brooks)
- Low level intelligence
  - No memory:
    - Can't handle sequences of behaviors
    - Can't learn anything
  - No world representations:
    - Doesn't know where it is
- Often more art than science

# HYBRID DELIBERATIVE/REACTIVE PARADIGM

- Reactivity by the end of the 1980´s was the major trend!
- But not remembering the state of the robot and the world is limiting!
- Question:
  - Should planning be reintroduced? How?
- Answer:
  - Hybrid systems!
  - More on this in the next AI course!

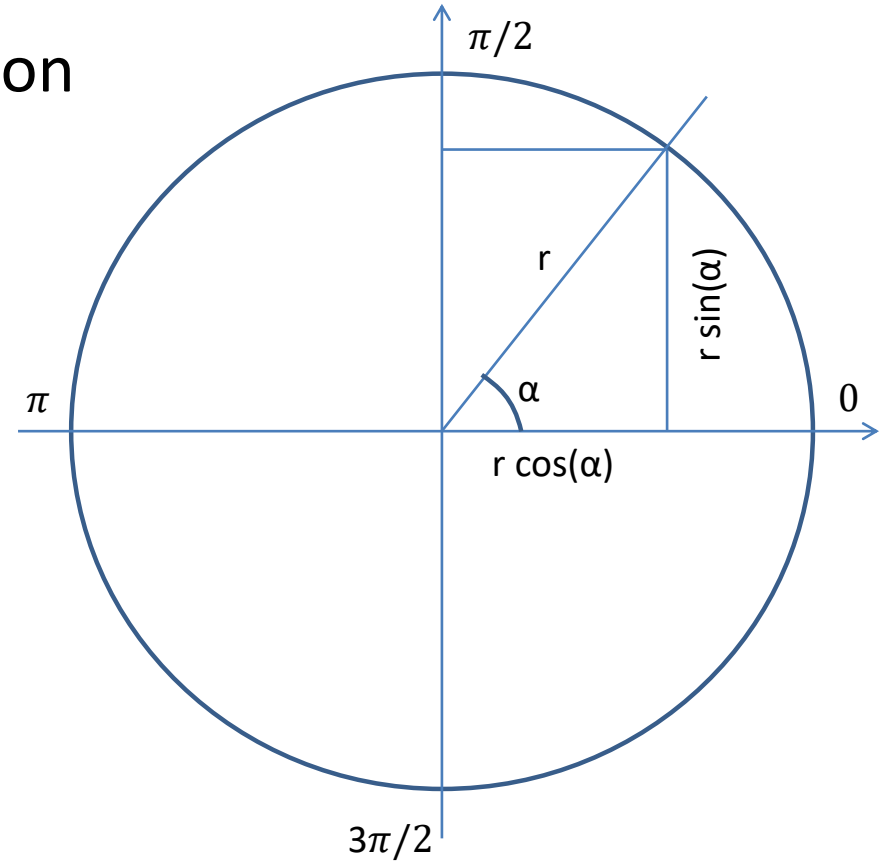
I've been talking to the computer

# Math for AI Robots

Thomas Johansson – thomasj@cs.umu.se
Dept of Computing Science
Umeå University

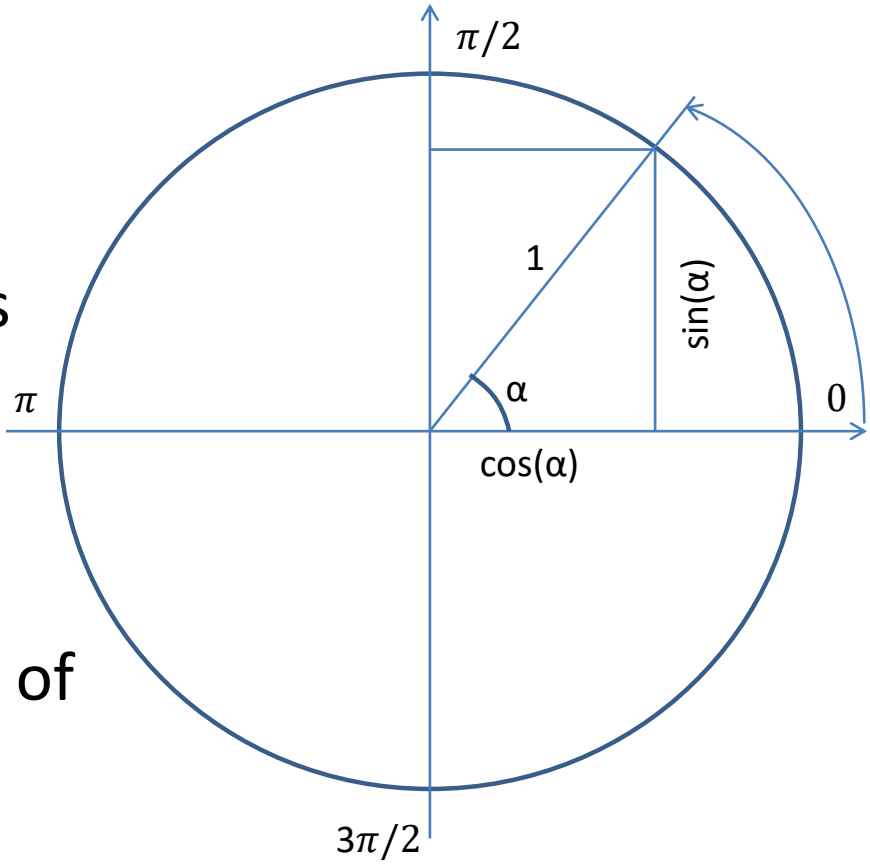
# Trigonometric Functions

- The most common
  - sin(α)
  - cos(α)
  - tan(α) = $\frac{\sin(\alpha)}{\cos(\alpha)}$
  - Inverses of the above
  - acos, asin, atan



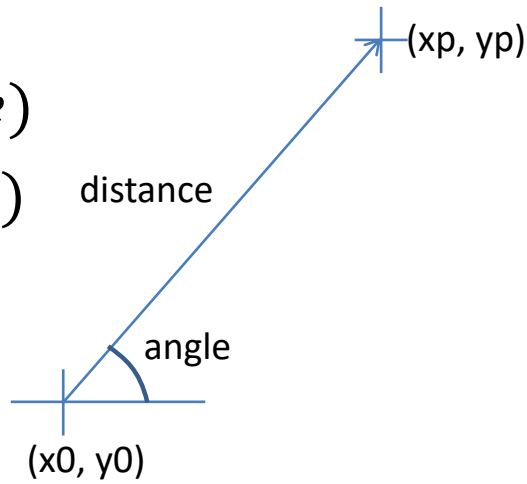
# The Unit Circle

- Radius = 1
- Angles increase anticlockwise
- Angles in radians
- One radian = 180/ $\pi$ degrees (about 57.3)
- Radians = length of arc



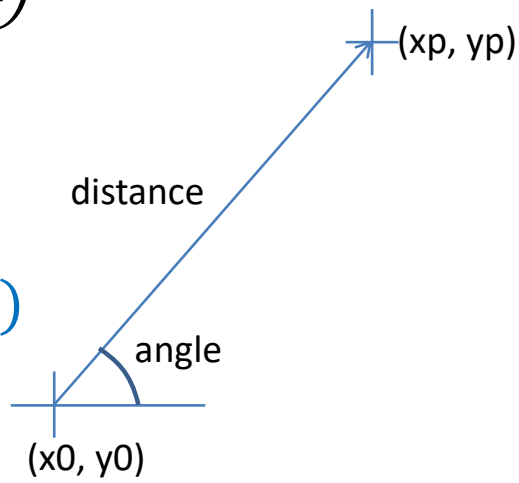
# Polar Coordinates

- Angle and distance instead of x, y
  - $Let\ dx = xp - x0\ and\ dy = yp - y0$
  - $distance = \sqrt{dx^2 + dy^2}$
  - $dx = distance\ * \cos(angle)$
  - $dy = distance\ * \sin(angle)$
  - $\frac{dy}{dx} = \tan(angle)$



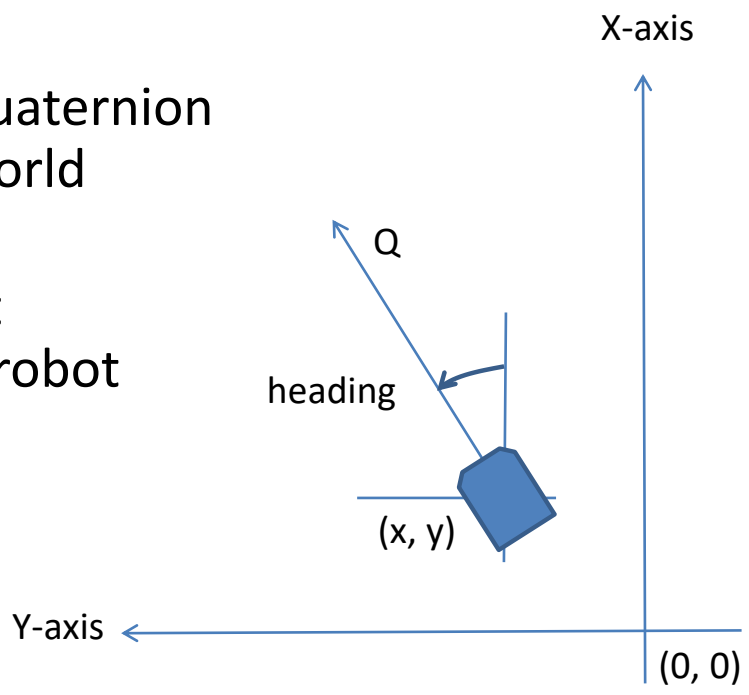
# Inverse Functions

- Angles from x, y
  - $angle = arcsin(dy/distance)$
  - $angle = arccos(dx/distance)$
  - $angle = \arctan(\frac{dy}{dx})$
  - But dx may be zero
  - Use $angle = atan2(dy,\ dx)$



# The Robot

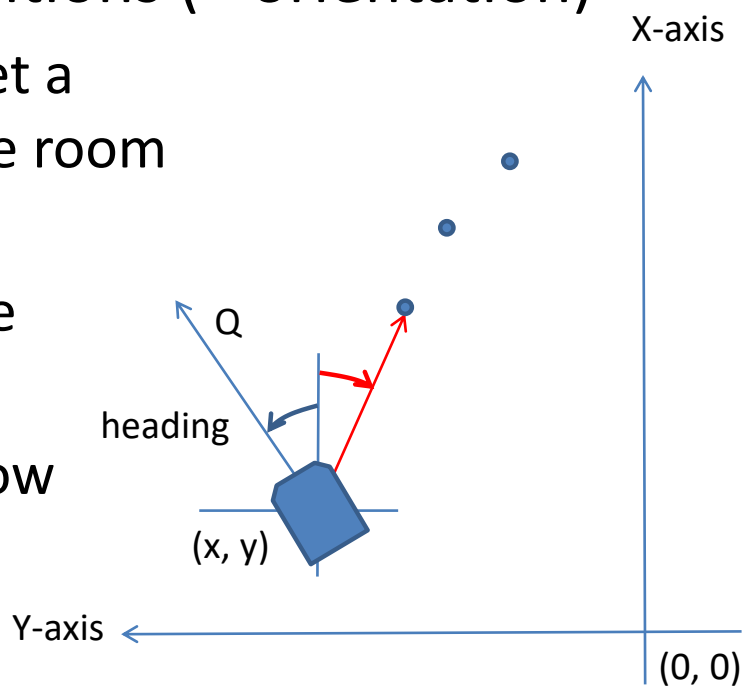
- What you get from the robot is
  - Its position, x, y, z
  - Its orientation, a Quaternion in relation to the world (the room)
  - From Q you can get the heading of the robot
  - Note the change of axes




  - $q = \cos(a/2) + i\,(x * \sin(a/2)) + j\,(y * \sin(a/2)) + k\,(z * \sin(a/2))$

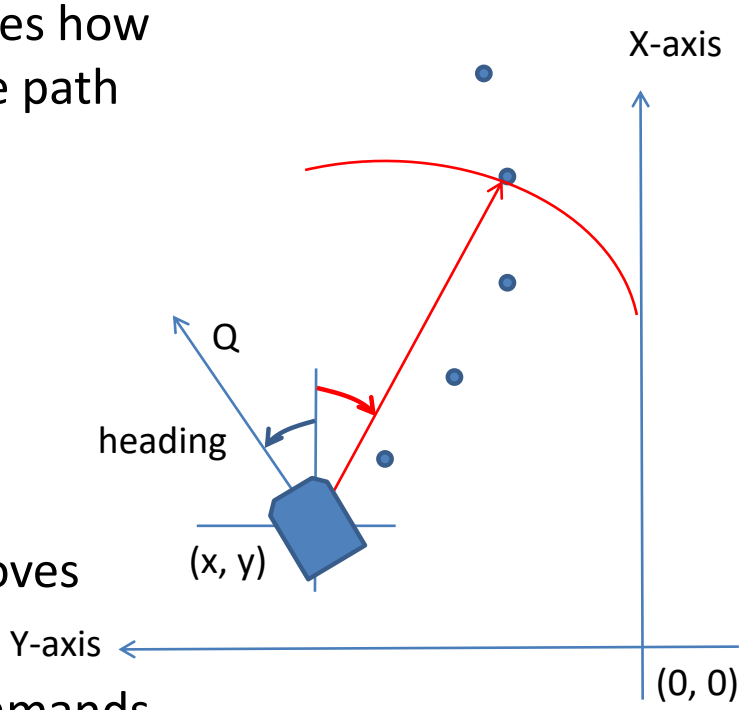
# The Path

- A sequence of positions (+ orientation)
  - From it you can get a set of points in the room
  - You must find the distance and angle to all these points
  - Then you know how to steer the robot



# The Algorithm(s)

- Choose *Follow the Carrot* or *Pure Pursuit*
  - You will have to choose a 'lookahead distance'
  - The size of this determines how accurately you follow the path
  - Long distance – faster but riskier
  - You should not visit every point exactly
  - Do not stop-turn-start, instead alter the speed and turn as the robot moves
  - Note: put a small delay (10-50 ms) between commands



# Localization

- A JSON object (string)
  - Returned from the robot
  - Stored in the path file in a JSON array
    [ {...}, {...}, ... ]
  - In Python – map to dictionary or similar
  - In Java – map to Map<String, Object> or Collection<Map< ...>>


```
{
  "Pose":
  {
    "Orientation":
    {
       "W":-0.70752808315921567,
       "X":0,
       "Y":0,
       "Z":0.70668522804785294
    },
    "Position":
    {
       "X":0.062024351309485075,
       "Y":-4.8787359764368405,
       "Z":0
    }
  },
  "Status":0,
  "Timestamp":75949220
}
```

# Fundamentals of Artificial Intelligence

## Machine Learning and Applications

Adam Dahlgren
Department of Computing Science

Umeå University

## Outline

- Overview of the machine learning process
  - Looking at code together
- Examples of interesting applications
- Some notes on deep learning
- General summary of current state of machine learning

## Rules during this lecture

- Whenever there is a question, at least one person must answer.
- I might lose track of space and time if interrupted
- But, whenever you do not understand, interrupt me.
  - You are most likely not alone
  - I might not understand either
  - Ask in Swedish if you are uncomfortable with English.
  - I want two volunteers that keep an eye on the chat in case I miss questions.
- Asking the right questions is as important as knowing the right answers.
- Remind me when you need a break

# Introduction

# Some repetition

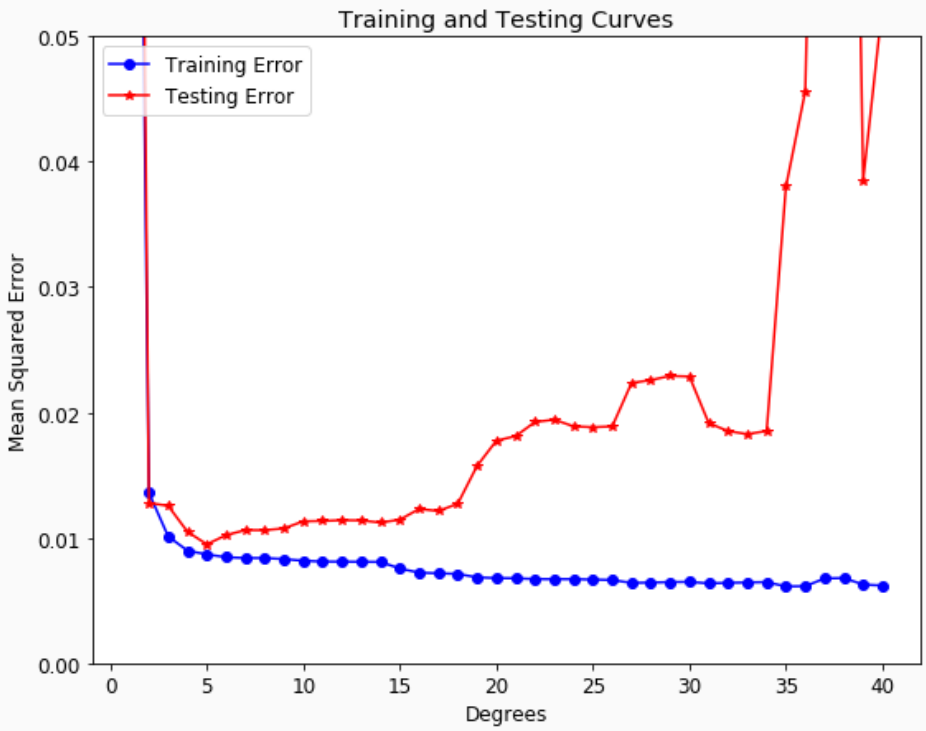



**Figure 1:** What is this an example of?

# The Machine Learning Process

## Recap

How do we define a machine learning?

A process that

- improves behaviour (performance)
- on a given task
- by experience

Can someone give me an example of another way of measuring performance than accuracy?

## Recap cont'd.

Define/explain

- Supervised learning
- Hypothesis space
- Overfitting
- Underfitting
- Ockham's razor
- Frame Problem

## Machine Learning Process - Overview

- It is a *highly* iterative process
- Usually requires lots of computational resources
  - GPUs
  - RAM
  - Time constraints
- Difficult to know how to improve performance
- Might be a hassle to actually fit the data and the model together
- Dependent on specialized software

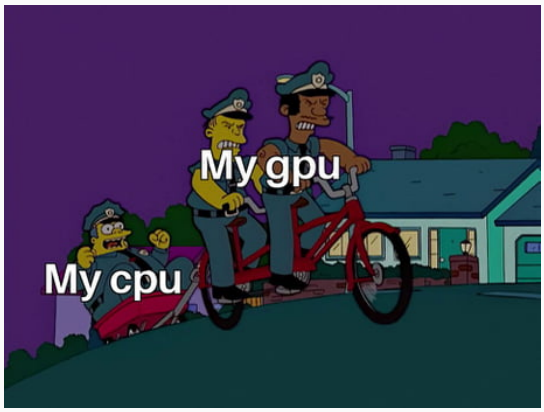



**Figure 2:** The usual setup.

## Software

We have a couple of established (open-source) libraries that we can use

- Python is a defacto standard for machine learning
  - Wrapper of C/C++ implementations
- Tensorflow, developed by Google and optimized against Nvidias GPUs
- Keras, built on Tensorflow, more high-level, contains all standard tools
- Pytorch, developed by Facebook
- Honorable mentions
  - Scikit-learn, R, Caffe, Weka, Numpy/Scipy, Theano, et c.
- Cloud based - Amazon, Microsoft, Google, IBM, Oracle

Your best bet is probably Keras in the beginning, but it becomes trickier when you move closer to the cutting edge.

## Repetition: An algorithm for learning

1. Collect a (large) set of examples
2. Divide them randomly into a training set and a test set
3. Generate a hypothesis using the training set (training our model further)
   - E.g. update the parameters of our model using gradient descent.
4. Measure the performance on examples from the test set (e.g. number of correctly classified examples).
5. Repeat until satisfactory performance is achieved.

This produces a model of the relationship between the input/output space that the classifier can use later.

# In its simplest form

```
from keras.models import Sequential
from keras.layers import Dense
from keras.utils import np_utils
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split

iris = load_iris()

(x_train, x_test, y_train, y_test) = train_test_split(iris.data, iris.target, test_size=0.3)

y_train = np_utils.to_categorical(y_train, 4)
y_test = np_utils.to_categorical(y_test, 4)

model = Sequential()
model.add(Dense(8, input_dim=4, activation='relu'))
model.add(Dense(4, activation='softmax'))
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
model.summary()

model.fit(x_train, y_train, epochs=100)

print(``Accuracy: '', model.evaluate(x_test, y_test)[1])
```

## How do we prioritize?

**Figure 3:** One soup (model) with too many chefs (things to tweak) requires a systematic approach.

## Traning- and test data

As we've seen, the way that we split our dataset can be paramount

- Training data - do we have enough?
- Validation data - what do we validate on during training?
- Test data - is it representative of the problem? How do we compare results?

When we have lots of classes, it is important to make sure that each class is sufficiently represented in each of these sets. How can we do that?

## Cross validation

Many of you will have noticed strange behaviour in your plots from the k-NN assignment.

- Why is it that the `random_state` you choose was so important?
- Give an example of how a bad selection of training-/test data could look like
- What can be done?

K-fold cross validation.

## K-fold cross validation

- Instead of choosing just one random division
- We split the dataset into $k$ parts, using each part as the test set once.
- We then report the average over all $k$ iterations as the accuracy or error.

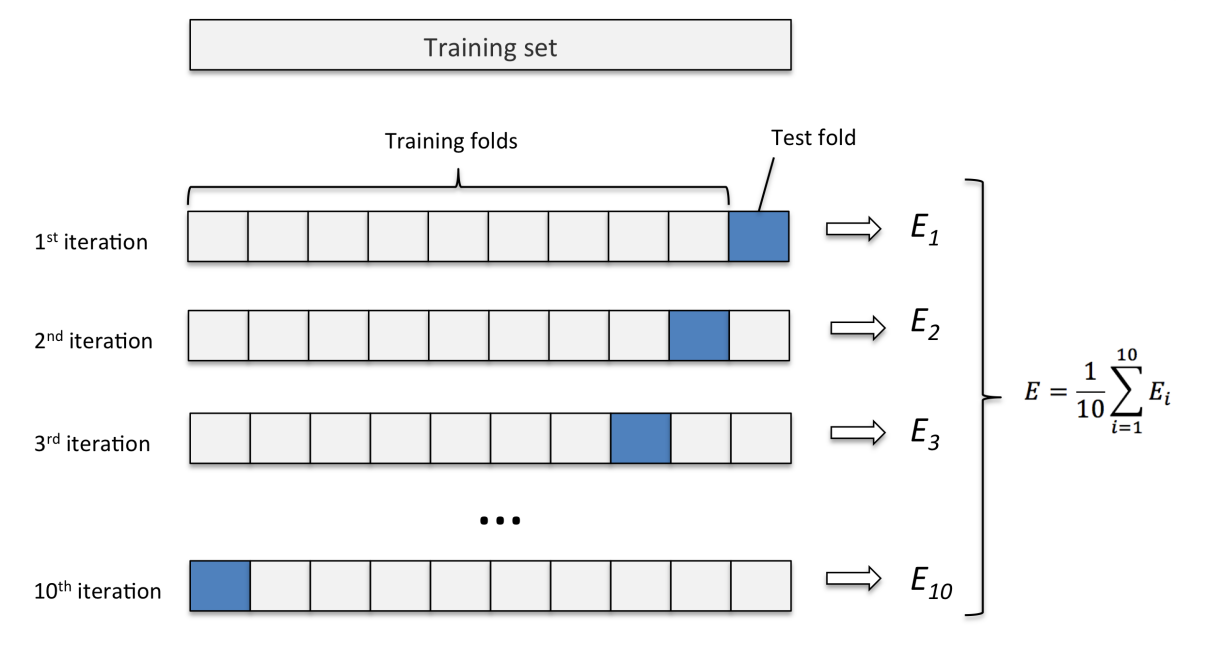



**Figure 4:** 10-fold cross validation. CC-license [1].

[1] http://karlrosaen.com/ml/learning-log/2016-06-20/

## Traning parameters

We have some parameters that are related to the training process

- Learning rate
  - How drastic can our parameter updates be? Step size in our optimization.
- Number of epochs
  - How many times should we go through the training data?
- Batch size
  - How many samples should we look at simultanously?
  - Useful to normalize over batches [1]

## Measuring performance

After each training iteration we want to measure performance

- Precision and recall can be defined per class
- Precision
  - Conceptually different from accuracy, although really similar
  - The number of correct predictions
  - Example where this is important?
- Recall
  - The fraction of total number in class correctly predicted
  - Example where this is important?
- Resources
  - RAM or disk consumption
  - Ecological footprint

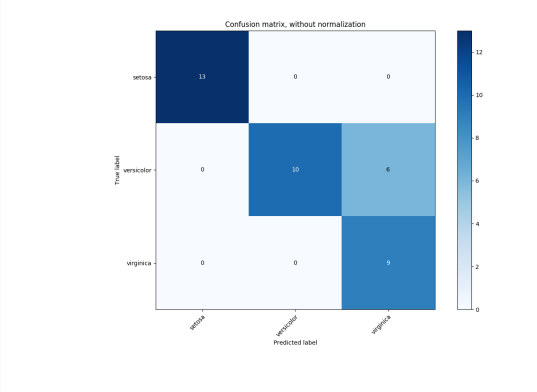



**Figure 5:** Example confusion matrix for Iris dataset.

# Confusion matrix

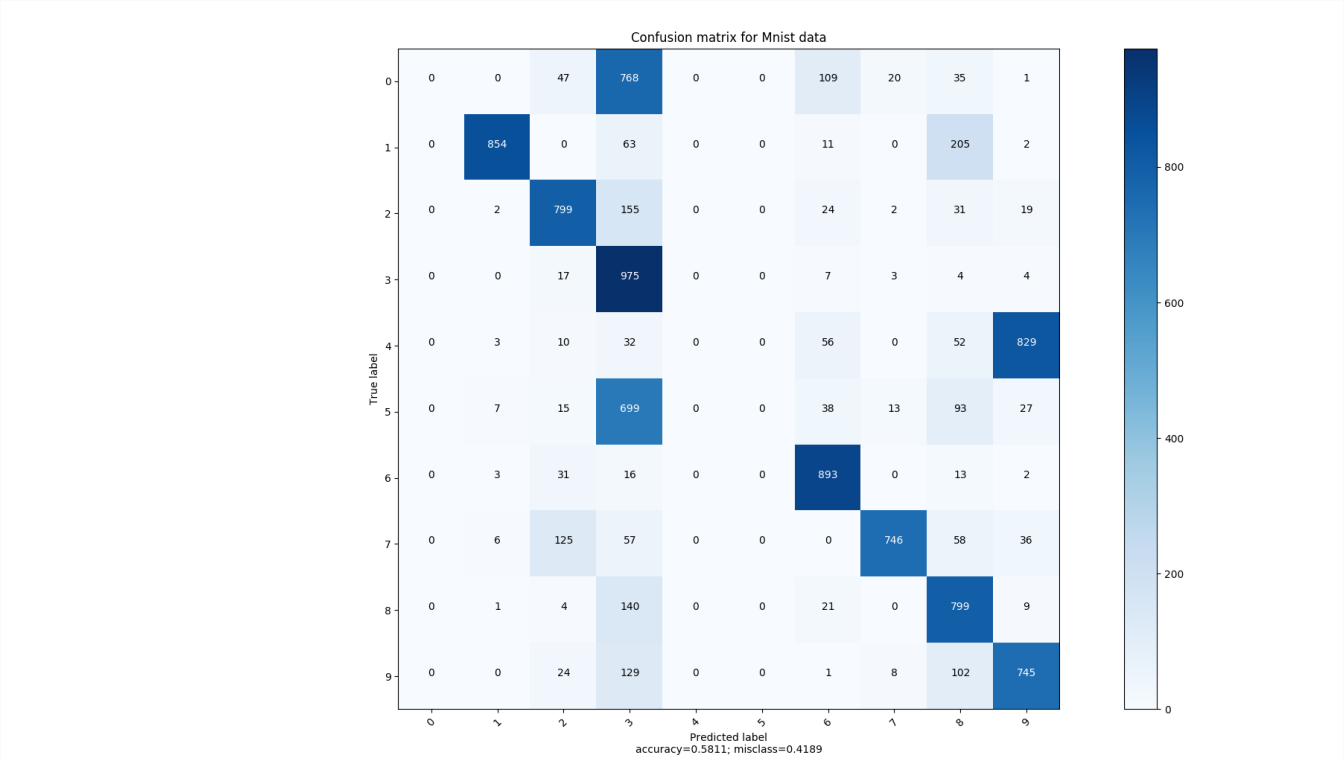



**Figure 6:** Confusion matrix for MNIST dataset.

## When should we stop?

Discussion: at what point should we stop training our model?

- Could we iterate until below a certain error?
- Always a fixed number of epochs
  - Can someone give me a reason why this can be bad?
- Early stopping
  - We want to avoid overfitting
  - We stop training after a number of training cycles has increased our test error

## Complexity of model

Can we increase the complexity of the model somehow?

- We can only vary the $k$ in k-NNs
- In regression we could choose a higher-degree polynomial
- In neural networks we can increase the number of nodes

# Deep learning

## Deep Learning

- Neural network comeback in seminal paper from 2012 on Convolutional Neural Networks [2].
- Fueled by advancements in hardware, it became possible to train much larger models
- Bengio, Hinton, and LeCun were awarded the Turing award for their work in this field
- Common saying: almost certainly the next best hammer to all nails

## Object detection and transfer learning

- Object detection is more or less a solved problem, with better-than-human performance [3]
- Has opened up for more complex tasks such as Visual Question Answering
- Transfer learning
  - Standard is to initialize our model with random values
  - With transfer learning we initialize with the parameters from an already trained model
  - The idea is that we only need to finetune to our problem
  - The previous training (for a different problem) should have picked up some general structure we can reuse
  - Especially useful if we have little data
- Transfer Learning common with models trained on ImageNet [4]

**Figure 7:** Example of tasks we are able to solve with deep learning.

## Generative models

So far we have mostly looked at supervised learning

- Can you give me three reasons why this is insufficient?
  - Labeling data is expensive, time consuming, and error prone
  - Unsupervised learning to understand underlying (*hidden*) structure
- Here we want to generate new examples given previous data
  - Generating new viewpoints
  - Completely new examples
  - Autocomplete in images (colorization, adding/removing segments)
- Many examples of really impressive applications in recent years

# pix2pix [5] - GAN example application

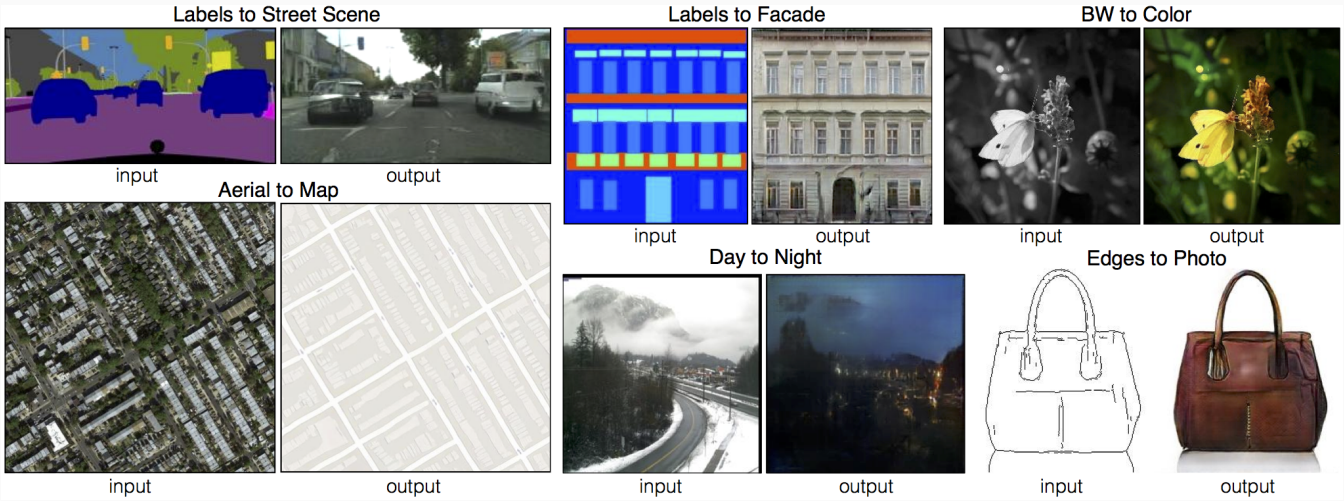



**Figure 8:** Example application built on GANs. Generate a realistic rendering of a given sketch. Take a look at Github [2].

[2] https://phillipi.github.io/pix2pix/

# Generative Adversial Networks

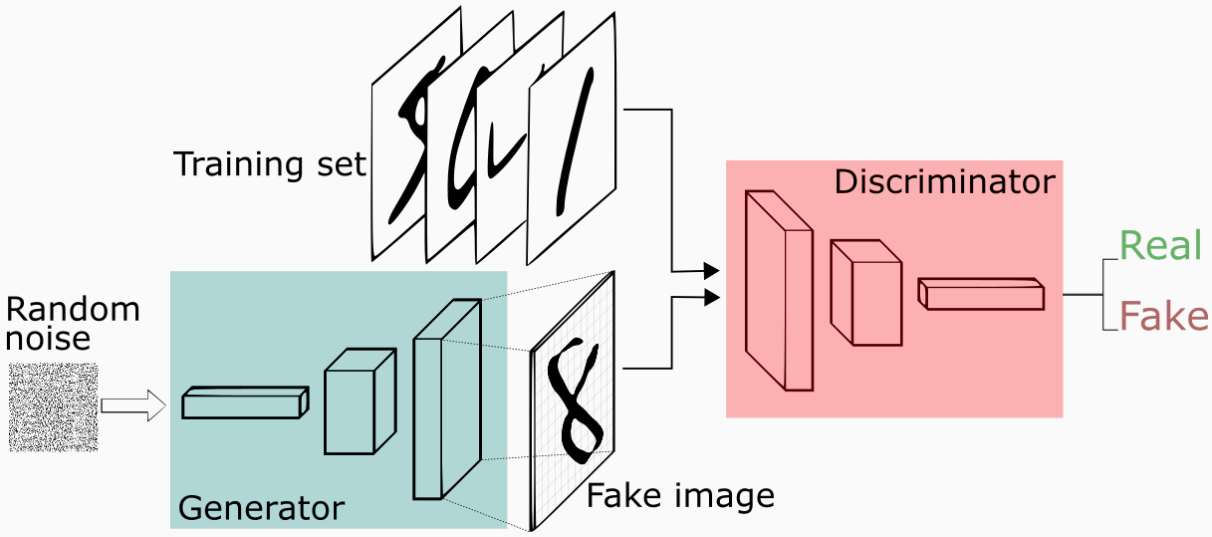



**Figure 9:** Overview of the general GAN architecture. Image credit: Thalles Silva.

## Example application with CycleGANs

CycleGANs introduce the idea that in image-to-image translation we want to do style transfer, not just generate good output.

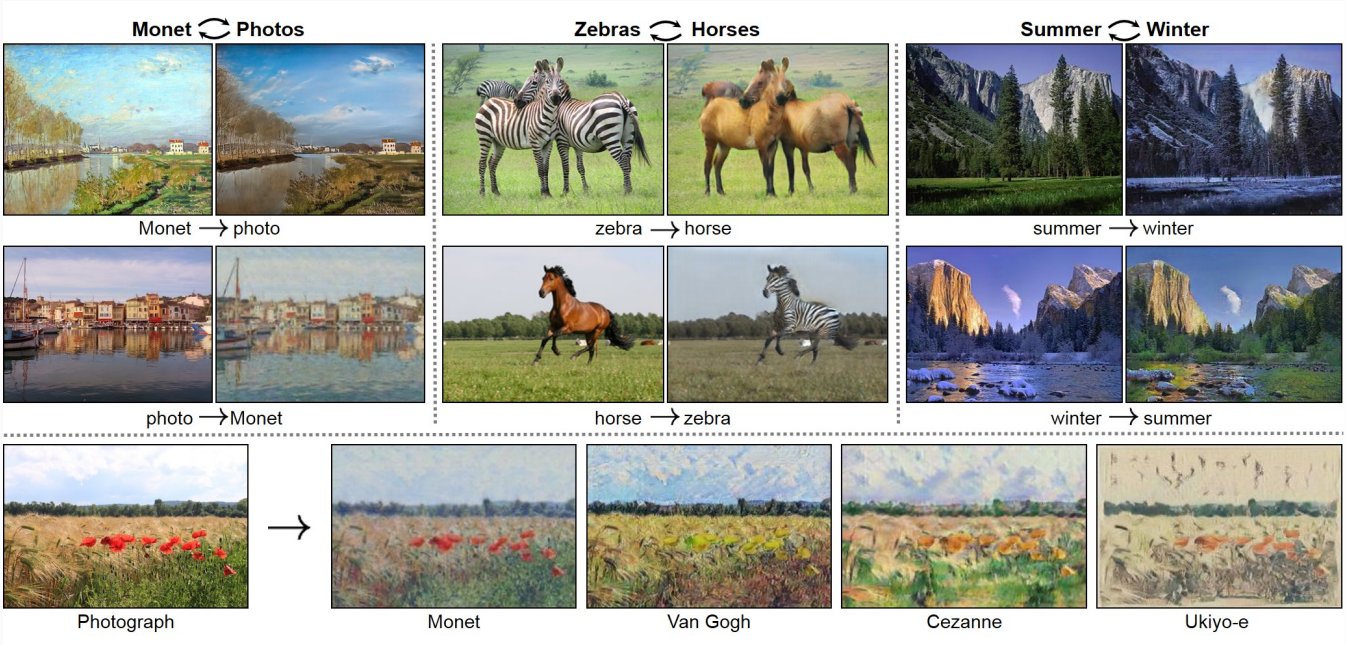



**Figure 10:** Image-to-image translation using CycleGANs [5].

## And sometimes they do

https://thispersondoesnotexist.com/

**Figure 11:** Artwork generated using GANs - sold for $432,500 in 2018 [3].

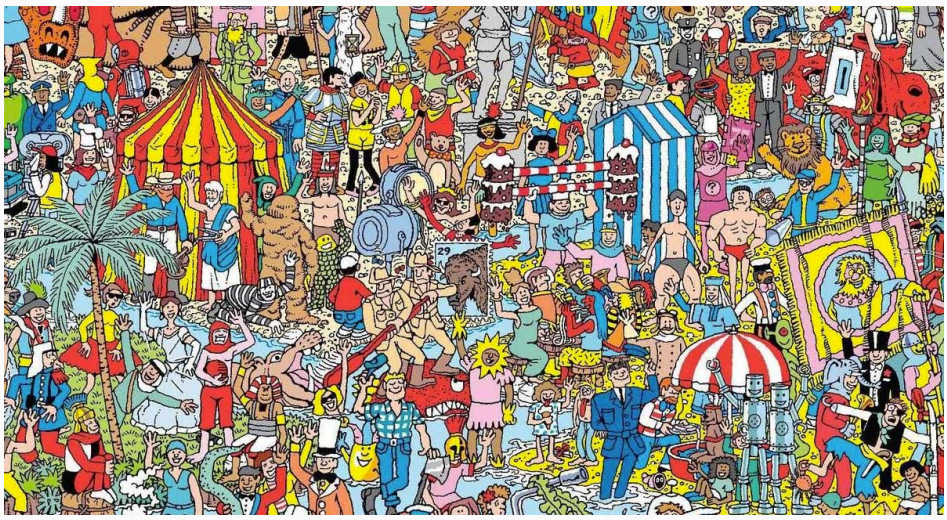
[3] https://edition.cnn.com/style/article/obvious-ai-art-christies-auction-smart-creativity/index.html

# The relationship between GOFAI and Deep Learning

## Kahneman's System 1 and System 2

An interesting parallel that many researchers relate to is the work by Daniel Kahneman and Amos Tversky that won them the Nobel prize in economics

- The idea is that we have a *System 1* that is fast, automatic, and unconcious
  - Riding a bike
  - 2+2=?
- *System 2* is slow, logical
  - Determine social behaviour



## Parallel to machine learning

The argument now is that AI systems can be modelled the same

- Machine learning models are highly specialized in narrow tasks, similar to system 1
- Good Old Fashion AI (GOFAI) such as logic, the more deliberate system 2
- A surge in interest in combining GOFAI and ML, see e.g. [6]

# Machine Learning Today

Some of the hot topics in the machine learning community

- Safe AI
- More interdisciplinary work
  - Important for explainable AI et c.
- Building models that need less data
  - Again, comming back to the Frame Problem
- Really large models (e.g. GPT3 from OpenAI) using unlabeled data

## Guacamole cat

- Neural networks is traditionally a black box, unclear what part of the data decides the outcome
- Or; Current deep-learning mechanisms are unable to link decisions to inputs

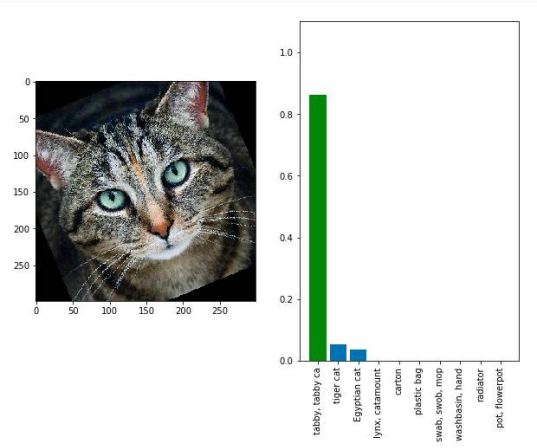



**Figure 12:** Image classified as a tabby cat.

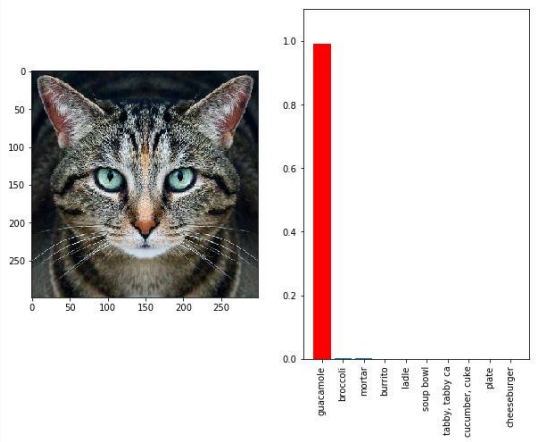



**Figure 13:** Image of the same cat, slightly rotated, classified as guacamole.

## Slightly more uncomfortable example

Could someone give an example that is more serious?

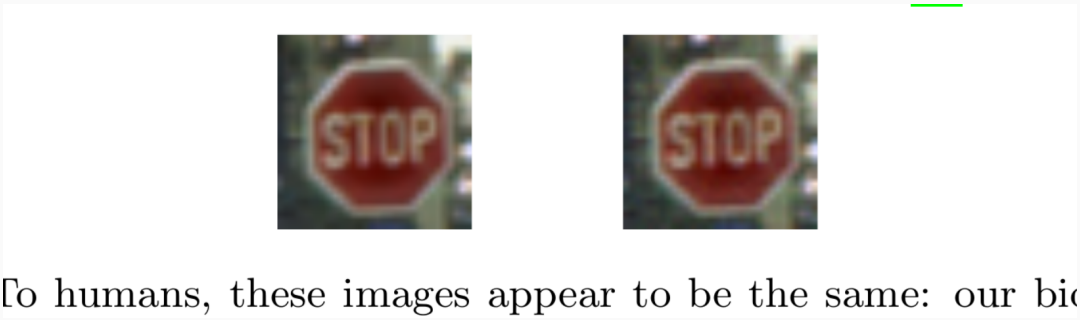



**Figure 14:** Two versions of the same image. One is recognized as a stop sign, the other as a yield sign [7].

The last decade has seen an increased need for what is known as Explainable AI (XAI).

## Theory of mind

Relationship between AI and philosophy

- Philosopher John C. Searles' distinction between *Strong* and *Weak* AI [8]
  - A system can: (1) *have a mind and mental states* or (2) *act intelligently*. Back to Chinese Room
- Traditional theory of mind is a framework within belief, knowledge, et c., are defined, and the ability to attribute this to oneself and others
- Rabinowitz et al. [9] has worked on a machine theory of mind, showcasing learning by observing
- The problem of other minds (see [10]): Partially the sollipsism argument, partially the problem of recognizing alien (as in unknown to us) intelligence

## Discussion

Is there anything you'd like us to elaborate on? Something we haven't talked about? Something we have talked about?

## Last slide

QUESTIONS?

## References i


- S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *CoRR*, vol. abs/1502.03167, 2015.
- A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25* (F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, eds.), pp. 1097–1105, Curran Associates, Inc., 2012.
- C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. E. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," *CoRR*, vol. abs/1409.4842, 2014.
- O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. S. Bernstein, A. C. Berg, and F. Li, "Imagenet large scale visual recognition challenge," *CoRR*, vol. abs/1409.0575, 2014.
- J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networkss," in *Computer Vision (ICCV), 2017 IEEE International Conference on*, 2017.

## References ii


- H. Geffner, "Model-free, model-based, and general intelligence," *CoRR*, vol. abs/1806.02308, 2018.
- N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami, "Practical black-box attacks against machine learning," in *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, ASIA CCS '17, (New York, NY, USA), pp. 506–519, ACM, 2017.
- J. Searle, *Mind, Language and Society: Philosophy in the Real World*. **MasterMinds series, Basic Books, 1999.**
- N. C. Rabinowitz, F. Perbet, H. F. Song, C. Zhang, S. M. A. Eslami, and M. Botvinick, "Machine theory of mind," *CoRR*, vol. abs/1802.07740, 2018.
- P. Godfrey-Smith, *Other Minds: The Octopus, the Sea, and the Deep Origins of Consciousness*. **Farrar, Straus and Giroux, 2016.**

# Fundamentals of Artificial Intelligence

## Introduction to Machine Learning

Adam Dahlgren
Department of Computing Science

Umeå University

## Outline

- Different types of learning
- Concrete example of supervised learning
  - k-nearest neighbour-algorithm
- Some learning theory (over-/underfitting)
- Introduction of Assignment 2

## Rules during this lecture

- Whenever there is a question, at least one person must answer.
- I might lose track of space and time if interrupted
- But, whenever you do not understand, interrupt me.
  - You are most likely not alone
  - I might not understand either
  - Ask in Swedish if you are uncomfortable with English.
  - I want two volunteers that keep an eye on the chat in case I miss questions.
- Asking the right questions is as important as knowing the right answers.
- Let me know if I start to mumble.
- Remind me when you need a break

# Introduction

## What is machine learning?

- Webster's Dictionary:
  *. . . modification of a behavioral tendency by experience.*
- What does this mean in our context?
  - Behaviour is the output of our system (e.g. actions for robots)
  - Experience is previously sensed data
- At what stage do we learn?
  - Online vs. offline learning

We want to modify the mapping from sensing to action.

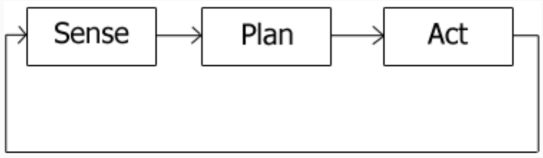



**Figure 1:** Traditional view of an intelligent agent.

## Machine learning: examples

Can someone give an example of machine learning in practice?

- Machine translation
- Data center management
- Computer-aided diagnostic systems
- Insurance premiums
- Control systems

What drives these applications? Data.

# Detexify

The tip of the day

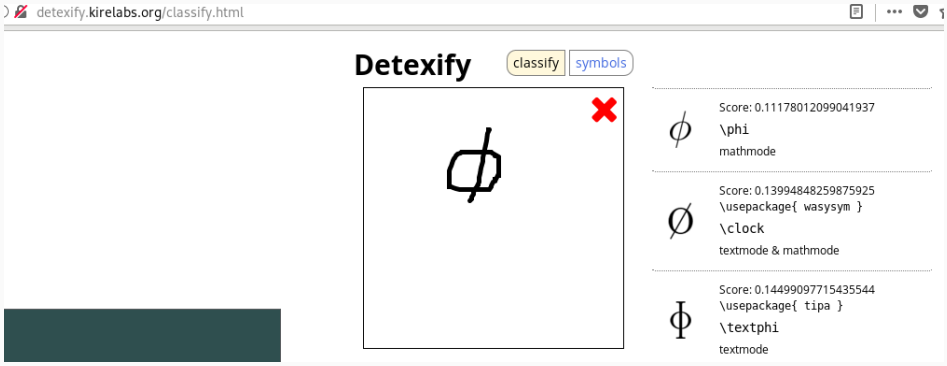



**Figure 2:** Example of a machine learning powered application

## Types of Learning

We have three main categories of machine learning

- Supervised Learning
  - Using a teacher to correct a classifier until it can handle given example input/output pairs.
- Unsupervised Learning
  - We have no labeled data and want to e.g. do clustering (recognize communities in social networks).
- Reinforcement Learning
  - Given e.g. an action, does that lead to a positive or negative outcome? Adjust accordingly, similarly to supervised learning. Common in robotics.

## Types of classifiers

We can define different types of classification tasks

- Binary classification
- Multiclass classification
- Regression
- Ranking
- Structured prediction

## Binary/multiclass classification

Given an input $x$, distinguish between two (or more) classes

- Binary classification
  - Determine whether an email ($x$) is spam or not ($y$).
  - Here $y$ is a discrete value yes/no.
- Multiclass classification
  - Given an image ($x$), what object ($y$) is shown? (E.g. cat, football, et c.)
  - Here $y$ belongs to a category.

## Regression/Ranking

We do not always want to label input with a certain class

- Regression
  - Given the temperature ($x$), predict the electric energy consumption ($y$)
  - Here $y$ is a continous output.
- Ranking
  - Given a number of matching queries ($x$), rank them ($y$) according to relevancy.
  - Mathematically speaking $y$ is a permutation applied to $x$.

## Structured Prediction

We usually have structure in input (relation between pixels, words in sentence, et c.).

- In structured prediction you also have structure in output.
- Given a sentence ($x$), label each word ($y$) with its role in the sentence.

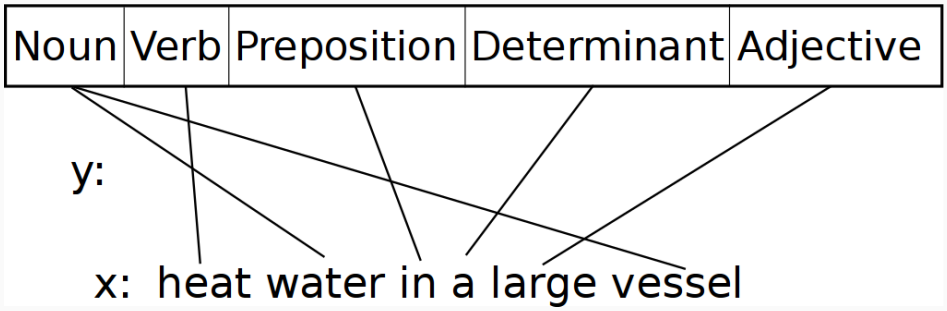



**Figure 3:** Part-of-Speech tagging

# Supervised Learning

## Supervised Learning

A set of examples is provided

$$(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$$

- Examples are divided in a *traning set* and a *test set*.
- Some true function $f : \mathcal{X} \to \mathcal{Y}$
  - We want a hypothesis $h$ such that $h \approx f$ (by looking at examples).
- Our hypothesis $h$ should generalize, i.e. perform well on both training data *and* unseen examples.
- The test set is used to estimate the generalized performance of $h$.

## Frame problem

**Q:** How does supervise learning tackle the frame problem?

## Learning theory

We have a couple of concepts

- Hypothesis space
  - E.g. all linear functions
- Empirical risk/loss minimization
  - Loss function that measures how well the prediction matches the true output.
  - Choose a hypothesis that minimizes this loss over all samples.
- Computational complexity
  - How complex is the relationship between the input- and output space?
  - How expressive is the classifier?

## An algorithm for (supervised) learning

1. Collect a (large) set of examples
2. Divide them randomly into a training set and a test set
3. Generate a hypothesis using the training set
4. Measure the performance on examples from the test set (e.g. number of correctly classified examples).
5. Repeat until satisfactory performance is achieved.

This produces a *model* of the relationship between the input/output space that the classifier can use later.

## Example: Curve fitting

Construct a $h$ that fits the traning set (i.e., $h$ is consistent with $f$ on all examples).

- We have a couple of samples $(x_i, y_i)$.
- We would like to fit a line that explains this data.
- This approximation should ideally generalize well.

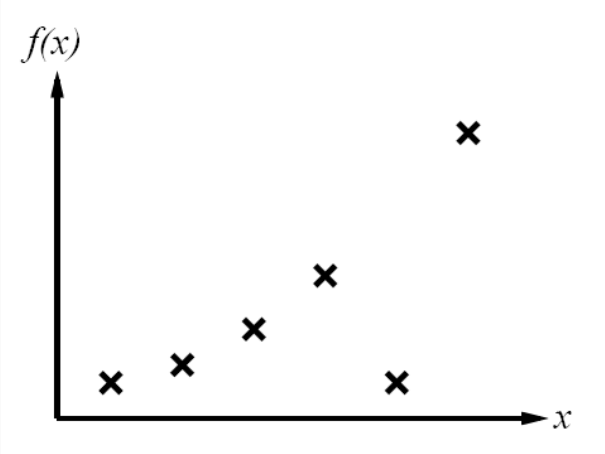



**Figure 4:** Some samples from a 2D data set

## Example: Curve fitting

Linear regression using e.g. the least squares method

$$h(x) = k_0 + k_1 x$$

$$S = \sum_{i=1}^{n} r_i^2$$

What about the bottom-rightmost sample?

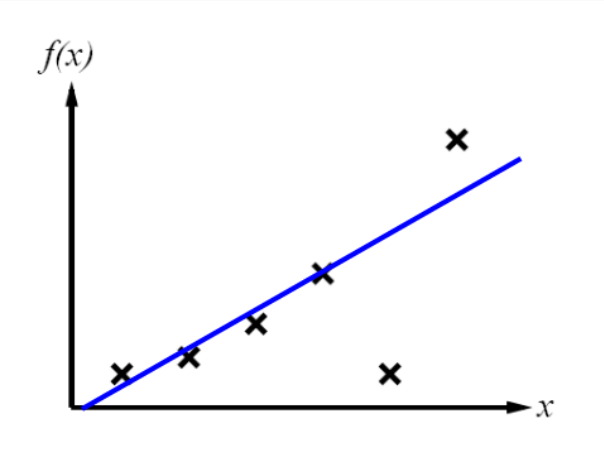



**Figure 5:** Linear fit to sample data

## Example: Curve fitting

We can try quadratic function as our hypothesis.

$$h(x) = k_0 + k_1 x + k_2 x^2$$

Given that the bottom-rightmost sample is just noise (or unseen), this hypothesis is a much better alternative.

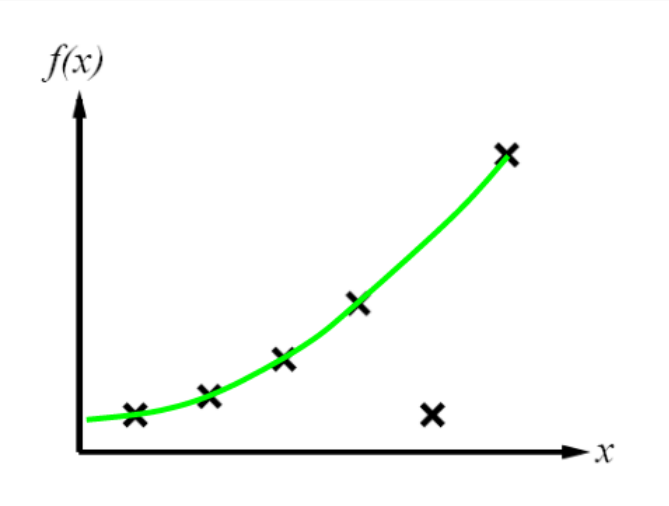



**Figure 6:** Quadratic hypothesis function.

## Example: Curve fitting

However, if the bottom-rightmost sample is legit data, we need a better hypothesis if we want to minimize the training error.

We can choose a $n$-degree polynomial

$$h(x) = k_0 + k_1 x + k_2 x^2 + \ldots + k_n x^n$$

Why might this be a bad idea?

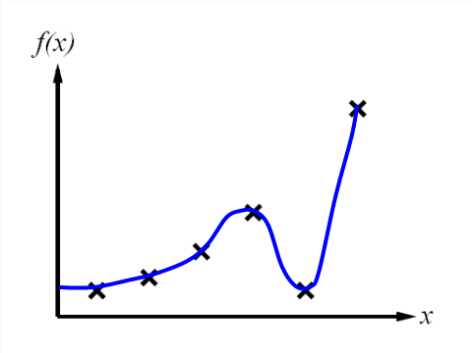



**Figure 7:** High-degree polynomial hypothesis.

## Example: Curve fitting

It is easy to construct a hypothesis function that fits the training data perfectly

**Ockham's razor**

- Maximize the combination of predictive accuracy with simplicity.
- Penalize too complex hypotheses, either implicitly or explicitly.
- We cannot only minimize the training error.

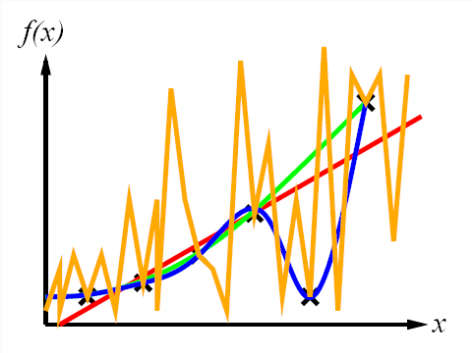



**Figure 8:** The orange zigzag fits the training data perfectly, but does it generalize well?

## Overfitting

In this example the polynomial classifier has a training error of zero, while the linear classifier has a non-zero error.

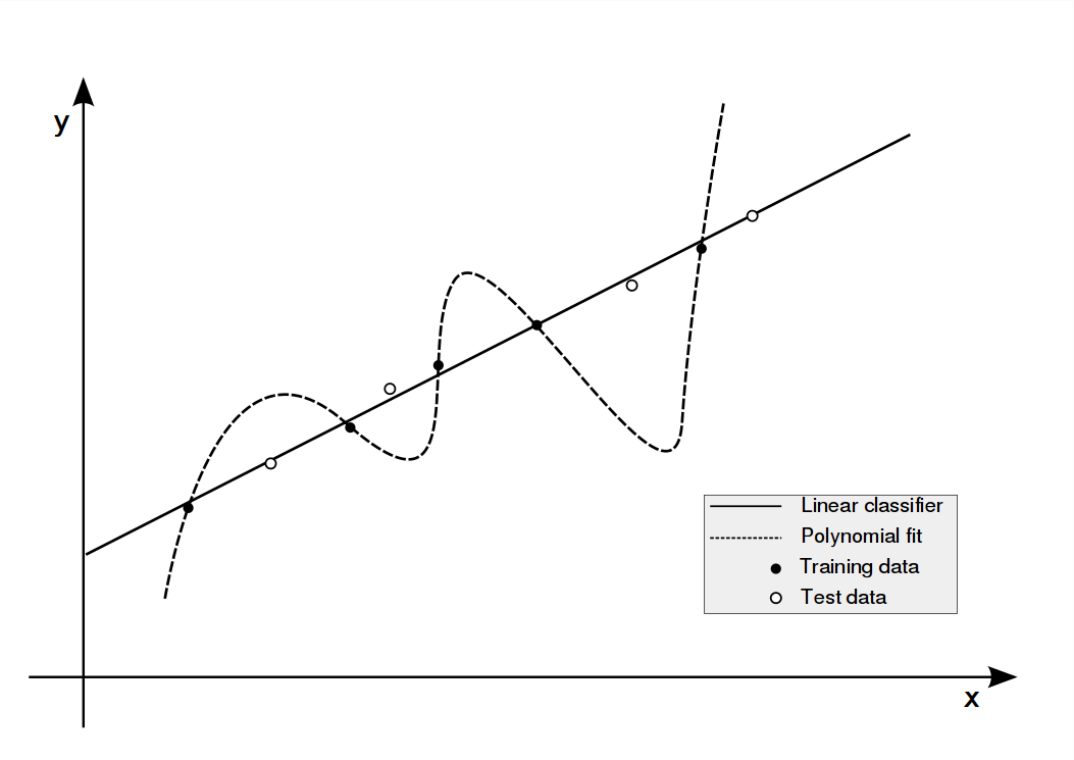



**Figure 9:** Example of overfitting

## Underfitting

Underfitting can occur

- when we have too little training data
- if our classifier cannot express the true complexity of $f$

We cannot generalize properly.

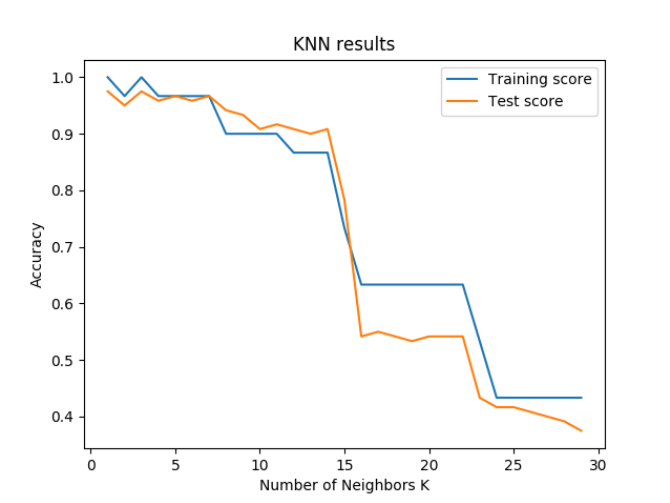



**Figure 10:** Example of underfitting

## Engineering features

In many situations, we want to do some feature extraction/selection

- We can build in domain knowledge into our machine learning algorithm
- What part of the data is really telling us something important?
- Is the original data on a usable form?
- Again, the frame problem
- Bias in AI and machine learning partially stems from this

What might be good features for predicting student success on a course?

## Feature engineering: example

Given a piece of text and the task to label it with a topic, what can we do to reduce the dimensionality?

- Remove common words like *is*, *the* with low information density
- Normalize words
  - Only lowercase
  - Word stemming - *manage* and *managed* $\rightarrow$ *manag*
- Application-dependent

## Feature engineering: example

Given a piece of text and the task to label it with a topic, what can we do to reduce the dimensionality?

- Usually represented as a feature **vector**
  - Mathematical properties that we can utilize
  - Requires a feature extraction function mapping from the input space to a vector space.

$$\phi(\text{"Two by two matrix"}) \rightarrow \begin{bmatrix} \vdots \\ 0 \\ 2 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ \vdots \end{bmatrix}$$

## Classification pipeline

Sentiment analysis of facial expressions in image

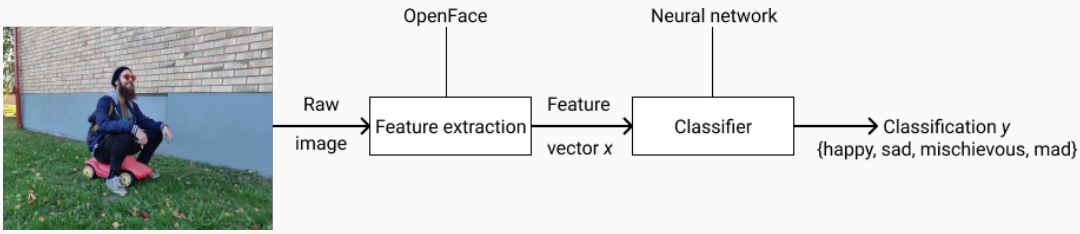



**Figure 11:** Classification pipeline for image to sentiment analysis

- Feature extraction could be made with e.g. OpenFace [1].
- The input space is the angle on eye brows, size of dimples et c.
- The output space is a set of predefined sentiments.
- Our classifier could be a neural network (as in your ANN assignment).

## k-Nearest Neighbours algorithm

Intuitively, given an unseen data point it should probably be classified as its closest neighbor

- This idea can be generalized to consider **k** neighbors.
- We want to learn a decision boundary that separates the classifications.

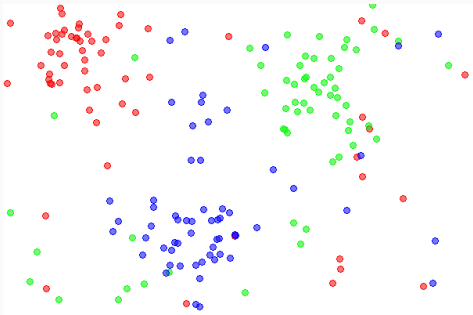


**Figure 12:** Data points plotted in the input space. Shared under Creative Commons [2].

## Visualization of k-NN decision boundaries

What happens if we consider too few neighbors?

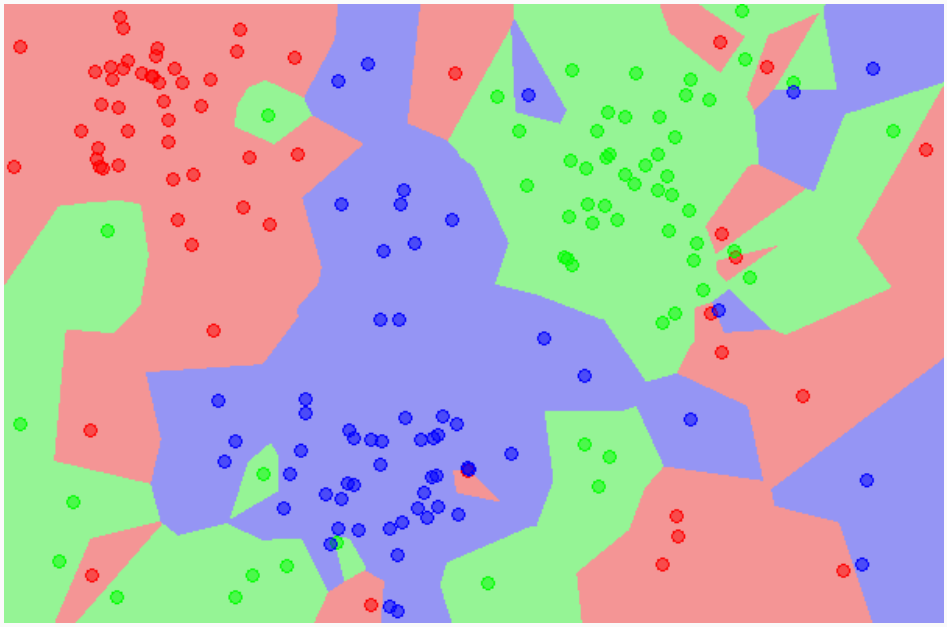


**Figure 13:** The decision boundaries of a 1-NN classifier. Shared under Creative Commons [3].

## Visualization of k-NN decision boundaries

Larger $k$ gives smoother decision boundaries

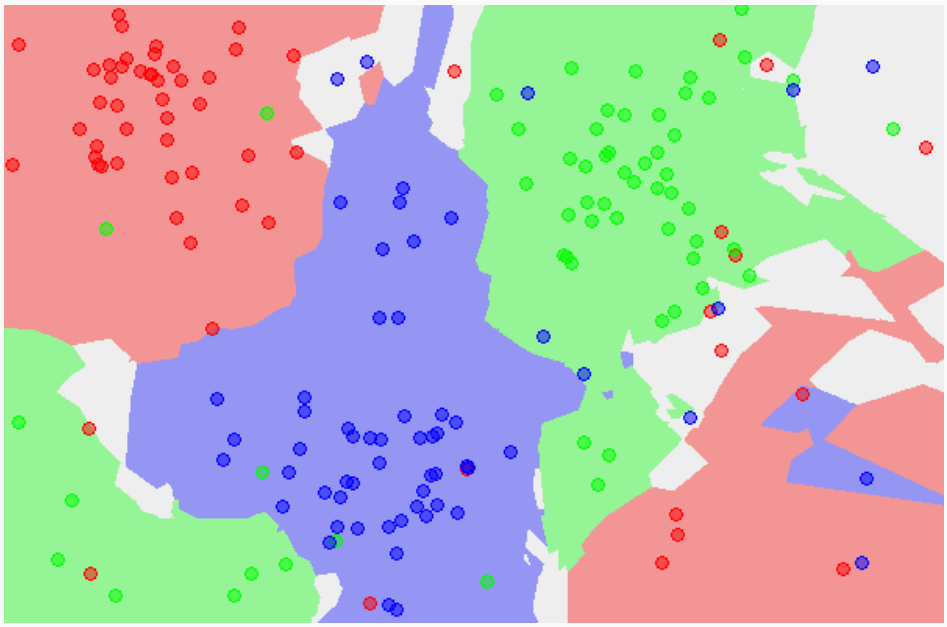


**Figure 14:** The decision boundaries of a 5-NN classifier. Shared under Creative Commons [4].

Can we consider too many neighbors?

## Complexity of the decision boundary

What is the optimal decision boundary?

- Upper- and lower bounds can be formulated using e.g. Kolmogorov complexity, Rademacher complexity et c.
- Complexity usually grows with dimensionality.
- The lower the ratio between the amount of data and the complexity, the harder it is to predict the boundary
- Related to the concept of entropy
  - What if labels are assigned at random?
  - This cannot be generalized

## The No Free Lunch theorem [5]

- *"Any two algorithms are equivalent when their performance is averaged across all possible problems"* - Wolpert, 1996.
- Without assumptions on the task, no classifier can be considered superior.
- There is no *best* classifier, choose like you choose your lunch.

# Unsupervised Learning

## Unsupervised Learning

We have no examples of input-output pairs

- The agent learns relationships between input and output
- Closely related to clustering

## Unsupervised Learning

Unsupervised learning techniques such as $k$-means clustering have an array of applications.

- Community detection in social networks
- Anomaly detection in computer systems
- Parallel to e.g. studies on personality traits in psychology

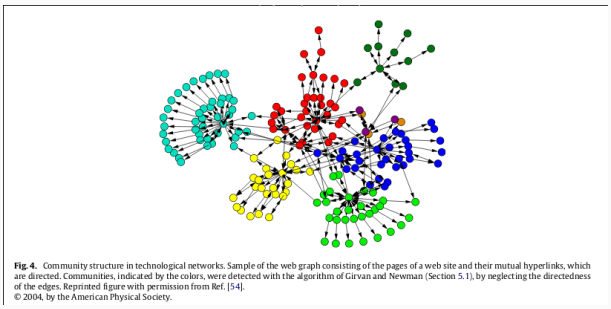



**Figure 15:** Clustering in social networks for community detection [6].

# Reinforcement Learning

## Reinforcement learning

Our last category of learning algorithms

- A correct answer $y$ is not provided for each input $x$.
- Instead, some sort of reward/scoring function is defined
- An agent explores the action space to determine valuable actions
- E.g. the sequences of actions that achieves the highest reward will determine the action $y$ in a game like chess.

## Reinforcement learning: example

Playing complex real-time strategy games has been proven really difficult. Recent advancements usually employ reinforcement learning.

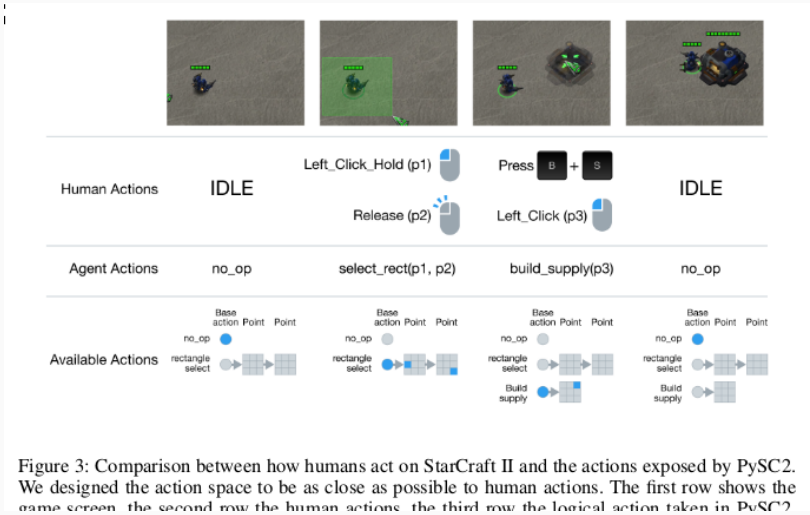



**Figure 16:** How people at Deepmind plays Starcraft II [7].

# Deep learning

## Deep Learning

We will not talk about deep learning today. Lets skip that until next lecture, and the ANN flipped classroom sessions.

# Assignment 3: k-NN

## Assignment 3: Overview

You should start working on the next assignment today

- You will train a k-NN classifier on two datasets
- Perform an analysis on the output with the theory from this lecture
- Write a neat little report on your findings
- You have a short deadline (one week), working in pairs
- You should **not** spend more than 10 hours on this assignment.

## Assignment 3: example

What can be said about Figure 17?

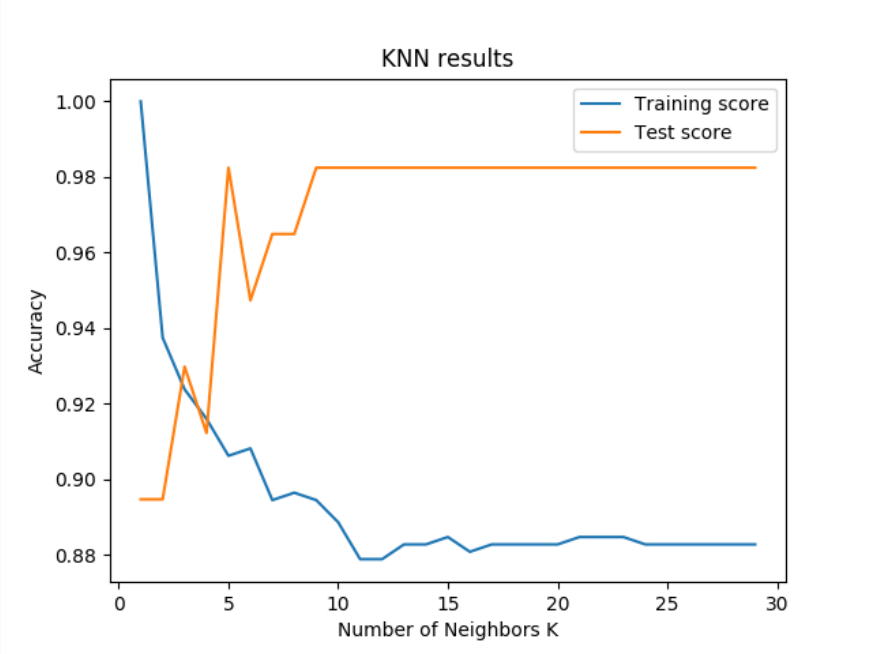



**Figure 17:** Example run of the given code.

## Last slide

QUESTIONS?

## References i


- T. Baltrušaitis, P. Robinson, and L.-P. Morency, "Openface: an open source facial behavior analysis toolkit," in *IEEE Winter Conference on Applications of Computer Vision*, 2016.
- W. Commons, "Data3classes," 2013.
  **File:** `Data3classes.png`.
- W. Commons, "Map1nn," 2013.
  **File:** `Map1NN.png`.
- W. Commons, "Map5nn," 2013.
  **File:** `Map5NN.png`.
- D. H. Wolpert, "The lack of a priori distinctions between learning algorithms," *Neural Computation*, vol. 8, pp. 1341–1390, 1996.
- S. Fortunato, "Community detection in graphs," *Physics Reports*, vol. 486, no. 3, pp. 75 – 174, 2010.

## References ii

O. Vinyals, T. Ewalds, S. Bartunov, P. Georgiev, A. S. Vezhnevets, M. Yeo, A. Makhzani, H. Küttler, J. Agapiou, J. Schrittwieser, J. Quan, S. Gaffney, S. Petersen, K. Simonyan, T. Schaul, H. van Hasselt, D. Silver, T. P. Lillicrap, K. Calderone, P. Keet, A. Brunasso, D. Lawrence, A. Ekermo, J. Repp, and R. Tsing, "Starcraft II: A new challenge for reinforcement learning," *CoRR*, vol. abs/1708.04782, 2017.

# Teleoperation and autonomy

Thomas Hellström
Umeå University
Sweden

Slide material contributions from Robin Murphy,Jussi Suomela



# How is a robot controlled?

1. By the human operator
2. Mixed human and robot
3. By the robot itself

Levels of autonomy!



# Levels of autonomy

**1a. Remote control**
- Visual contact / no sensor feedback

**1b. Tele-operation** – OCU provides sensor data
- Simple t-o: control of individual joints, motors etc.
- User space t-o: motion primitives e.g. internal closed loop velocity control of vehicle
- Safety-guarded t-o: e.g. emergency stop

**2. Semi-autonomous** (supervisory) control
- Shared control
- Traded control

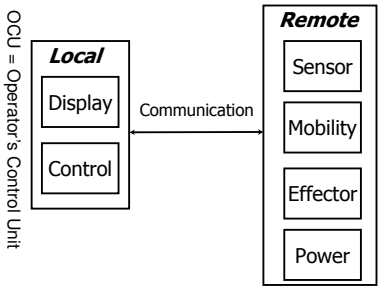
**3. Autonomous robots** – not here yet



# Remote control

- Not only toys
- The operator has most of the time straight visual contact to the controlled target
- Control commands are sent electrically by wire or radio

# Components of a Teleoperated system

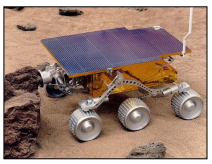





# Teleoperation Applications

Space
- Perfect for teleoperation: safety and costs
- Problems with very long delay

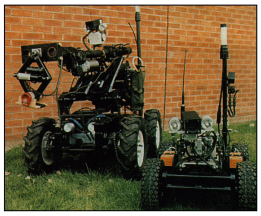
Lunokhod 1 (Луноход) "moon walker"
First t-o vehicle on the Moon 1970



Sojourner, fist t-o vehicle on another planet. Landed on Mars 1997

## Teleoperation Applications

- Military
  - underwater
  - ground
  - air
  - semiautonomous / internal closed loop control
- Anti terrorist
  - typically internal closed loop control

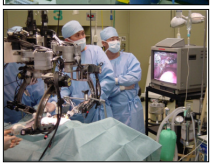




## Teleoperation Applications

Medical

- Endoscopic surgery
  - Surgery through small incisions or natural body openings
  - minimal damage, smaller risks
- Telesurgery
  - Surgeons can work over distances





## Teleoperation Applications

Mining

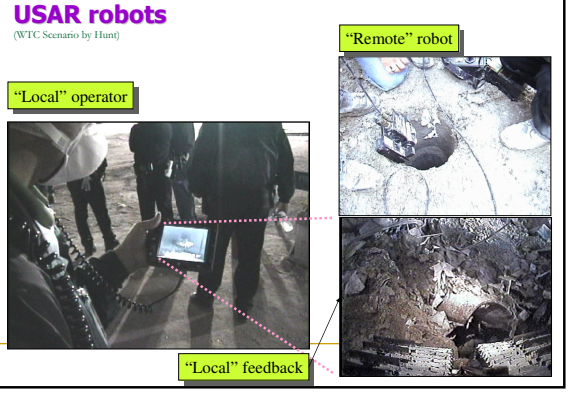
- Unsafe areas
- Cheaper operation

All images from http://www.cse.usf.edu/pages/ldp/ccd
Pictures chosen for pedagogical purpose. Two different robotic systems are shown

## Teleoperation Applications

**USAR robots**

(WTC Scenario by Hunt)




## Problems with Tele-operation

## Problems with Tele-operation (Murphy after 9/11)

- Lighting conditions
  - High variation in ambient light makes computer vision tasks difficult
- No "tactile" feedback
  - Couldn't really tell when the robot was stuck or when it was free
  - Robot didn't have proprioception (internal sensing)
  - Operator didn't have an external view of the robot itself
- Communications
  - High dropout rate after about 10 feet away!

## Simulator Sickness

- Common in Teleoperation
- Similar to motion sickness, but can occur without any actual motion of the operator
- Symptoms: apathy, general discomfort, headache, stomach awareness, nausea...



## Simulator Sickness

- Caused by *cue conflict*
- In cue conflict different nerves get different information from the environment
- Typically conflict between visual and vestibular inputs
- Especially when HMD is used and the time lags in vision and control

throw up

## Delays

- Acceptable control loop times
  - Nyquist sampling theorem: measuring frequency > 2 x system frequency
  - In practise (mobile machines):
    - < 0.1s : perfect
    - < 0.5 s : ok
- Delays depend on
  - Transmission speed (max. 300 000km/s)
  - System delays
- Long delays cause *Cognitive fatigue*

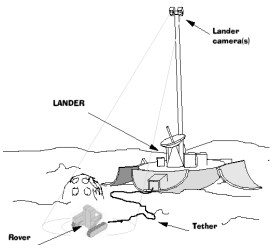


## Not really unmanned...

4 people to control it
(52-56 weeks of training)
- one for flying
- two for instruments
- one for landing/takeoff
- plus maintenance, sensor processing and routing

## Long delay teleoperation

- Earth-Moon-Earth: 2 seconds
- Earth-Mars-Earth: 37 seconds
- No possibilities for external closed loop control with a moving robot
- Instead: "move and wait" teleoperation






## Tele-operation

+ Doesn't depend on machine intelligence
+ Doesn't depend on a present operator
- Depend on good communication
- Hard for the operator ⇒
  - Cognitive fatigue
  - Simulator sickness
  - Many operators required

## Tele-systems Best Suited for Tasks:

- that are unstructured and not repetitive
- that require dexterous manipulation, especially hand-eye coordination, but not continuously
- that require object recognition or situational awareness
- that don't need display technology that exceeds bandwidth and time delays limitations of the communication link
- where the availability of trained personnel is not an issue



## Ways to improve Tele-operation

- Improve the HRI ⇒ less demanding for operator: TELE-PRESENCE
- Make the robot more intelligent ⇒ less demanding for operator and communication system:
  SEMI-AUTONOMY



## Tele-presence (remote presence)

"Virtual reality"

Provide sensory feedback such that the operators feels they are "present" in robot's environment

- Ideally all human senses transmitted
  - Vision, hearing and touch
  - Smell and taste
  - Balance, motion
- Demands higher bandwidth
- Less problems with
  Cognitive fatigue and
  Simulator sickness



## Vision

- Humans get 90% of their perception through vision
- "To see is to believe"
- Eyes are very complex opto-mechanical systems
  - FoV is (H)180 deg x (V)120 deg
  - Focused area only few degrees
  - Movements over whole area
- Extremely difficult system to be imitated



## Interface with Vision

- Head tracking
- HMD ⇒ relatively good feeling of presence

## Hearing

- Human range 16 – 20000Hz
- Important in telepresence
- Noise can be filtered out

## Touch & Force

- Tactile information ("touch")
  - mechanoreceptors – activated by pressure on the tissues
- Kinesthetic information ("force")
  - sense of position and motion of limbs and associated forces conveyed by receptors in the skin around the joints, tendons, and muscles, together with neural signals



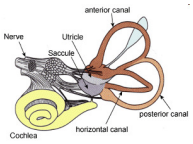
## Interface with Haptic feedback

- tactile sensing of the robot manipulator is fed back to the fingers of the operator

## Interface with kinesthetic (force) feedback

- Force is fed back to the operator
- Generates a real response in gripping and manipulation tasks
- Also in virtual environments

## Vestibular sensors




- Located inside the inner ear
- Responds to
  - Angular acceleration (and thus rotation)
  - Spatial orientation
  - Linear acceleration in the horizontal and vertical plane, i.e. to gravity
- ⇒ pose and movements of the head are detected



## Vestibular feedback

- Usually not needed in teleoperation
- Expensive to implement
- Usually in simulators to create presence
- If vision and vestibular sensors mismatch => simulator sickness

## Better than the real thing: Augmented reality

- Real information (usually image data) is mixed with additional virtual information
- Numerical information, real-time models, etc.

## Tele-presence applications

- Lawn mower
- Tele conferences
- "Taking care of" elderly
- Baby sitters
- Home robots
- Security
- Garden clubs



## Ways to improve Teleoperation

- Improve the HRI => less demanding for operator : TELE-PRESENCE
- Make the robot more intelligent ⇒ less demanding for operator and communication system : SEMI-AUTONOMY



## Semi-autonomus control

- General idea:
  - Teleoperation for hard tasks
  - Autonomy for simple tasks
- Reduces cognitive fatigue/ simulator sickness
- Demands lower bandwidth
- Less sensitivity to delays

Two major types:
- Shared control
- Traded control



## Shared control

The human operator
- Delegates a task
- Monitors the process
- Interrupts for hard sub-tasks, and if anything goes wrong

Two parallel control loops (the human and the robot control different aspects of the problem):
1. Autonomous (high intensity)
2. Teleoperated (low intensity)



## Shared control

**Example** (space robotics)
Task: *Release the bolts on shield H34.*

• Autonomous motion to shield H34.
• The human monitors and may interrupt if the situation becomes unsafe.
• The human releases the bolts by tele-operation

Note: Constant monitoring needed



## Traded control

- The human operator
  - Initiates action
  - Neither monitors nor interrupts
- If the operating conditions go outside the abilities of the robot, control is transfered to the human
- When the human takes over, she has to quickly acquire *situational awareness*
- When the robot takes over, it has to quickly acquire *situational awareness*

## Situational awareness

Most often refers to the operator's perception of the world

Important for
- pure teleoperation
- operator take over in semi autonomous systems: Low awareness ⇒ longer take-over time

Three levels of situation awareness (Endsley 2000):
1. there is perception of the relevant status information
2. there is comprehension of the status information
3. there is prediction, i.e. the ability to use this comprehension to consider future situations



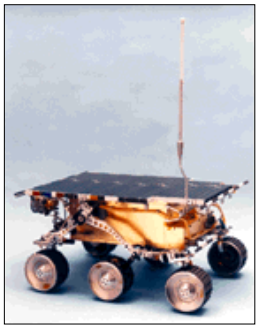
## Situational awareness

Experiences of robotic rescue researchers at the WorldTrade Center (Casper 2002):

- 54% of the time spent was reported to have been wasted trying to determine the state of the robot
- The operator gets confused by the egocentric camera view regarding Attitude (roll, pitch)



## Traded control - Sojourner



The first Mars rover, launched in December 1996. Landed on the surface of Mars on July 5, 1997.

11.5 kg

630 x 480 mm

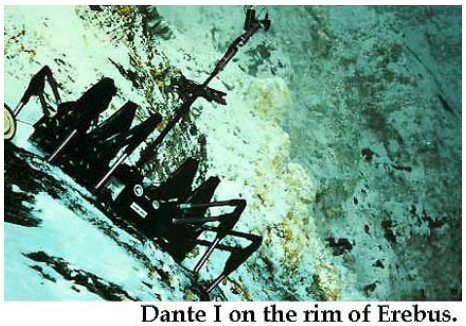
Worked by Teleoperation and semi-autonomous control

Example: "DRIVE TOWARD THAT STONE"

Sojourner avoids obstacles on the way



## Dante I

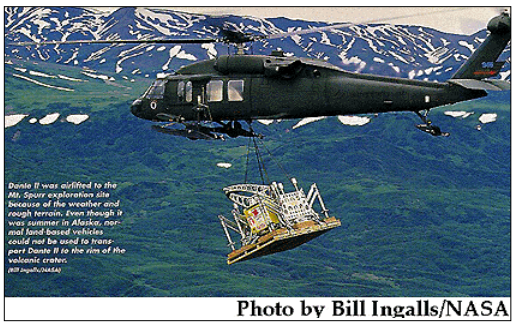


Dante I on the rim of Erebus.



## Dante II

## Dante II



## Interface design

## Interfaces

- Between the operator and robot/vehicle
- Strong connections with HMI and HCI, but additional problems
- As usual:
  The user interface is absolutely critical
  - make up 60% of commercial code

## Interface layout

depends on
- The level of autonomy
- The level of sensing/perception

## Levels of autonomy (again)

**1a. Remote control**
- Visual contact / no sensor feedback

**1b. Tele-operation** – OCU provides sensor data
- Simple t-o: control of individual joints, motors etc.
- User space t-o: motion primitives e.g. internal closed loop velocity control of vehicle
- Safety-guarded t-o: e.g. emergency stop

**2. Semi-autonomous** (supervisory) control
- Shared control
- Traded control

**3. Autonomous robots** – not here yet

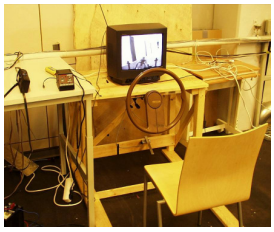


## Interface - Remote control

- No sensor feedback
- Low bandwidth

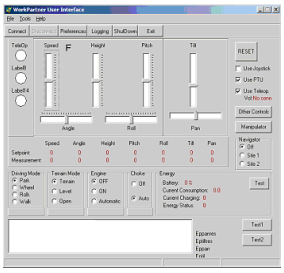
## Interface –Simple Tele operation

- "Direct" tele-operation
- Same view as onboard
- External closed loop control of motor speeds, height, ...
- Operator controls with hand controllers (like onboard)
- Realtime operator decision making is necessary
- High bandwidth, low delay communication



## Interface – User-space Tele operation

- Multimodal/multisensor
- Integrated display with combined sensor information
- Internal control-loops for speed, height, …
- (Autonomous safety functions)



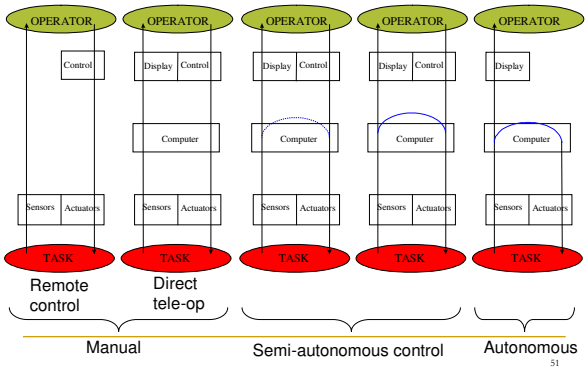


## Interface – Semi-autonomous Control

Support for

- high-level commands
  - Move to …
  - Grip
  - Look for …
- monitoring of success/errors
- Interruption of tasks

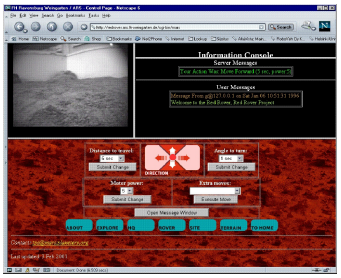
## Control methods (Sheridan 2003)





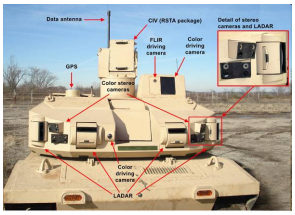

## Novel interfaces

- "novel" is relative
- gestures
- gazes
- brainwaves
- muscle movements
- WEB interfaces
  - multimodal
  - supervisory





## The Black Knight




Black Knight's sensor systems

http://www.youtube.com/watch?v=hRDs__6dFsE&feature=related



## The Black Knight's OCU

- Objects that are detected are overlaid on the driving map enabling drivers to maneuver around them
- Can plan paths to be manually driven by its operator
- *Guarded teleoperation:* The vehicle stops when it detects lethal obstacles in its path.

Black Knight's Robotic Operator Control Station (ROCS) displays images from the vehicle's driving cameras and includes a game controller-style hand controller and a touch screen interface for maps and vehicle systems.

## The Remote Robotic Reconnaissance Vehicle (R3V)

- Enhanced situational awareness using fused sensors

The robotic vehicle with a FLIR (forward looking infrared) and a low-light camera

Operator Control Unit (OCU), for control and display

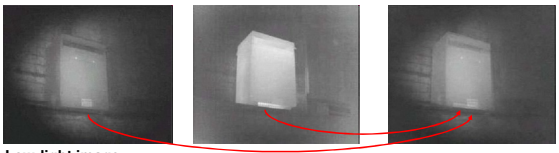


## OCU – Operators Control Unit

- Vehicle status and remote video via a 1024x768 LCD display
- Vehicle control: Speed , Steering
- Camera control: zoom camera, fader controls and camera tilting, manual iris, focus and gain control.



## Fusion of IR and camera



**Low-light image** **IR image** **Fused low-light and IR image**



## Assessing the usability of a HRI

- **Effectiveness**: the percentage of a task that the user is able to complete
- **Efficiency**: depends on the time needed to complete a given task
- **User satisfaction**: subjective

The three measures are weighted:

- **Life critical applications**: more weight to the effectiveness
- **Time critical applications**: more weight to efficiency
- **Entertainment**: more weight to user satisfaction



## Camera display modes

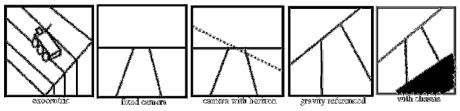
Three basic ways to monitor a robot's location, orientation and the world around it

- **Egocentric**: Inside-out perspective; "Through the windshield"
- **Exocentric**: Outside-in perspective; Radio-controlled planes.
- **Mixed perspective**: Inside-out perspective but includes information about orientation, e.g. artificial horizon displays

(Wang, Levis, Hughes 2004)



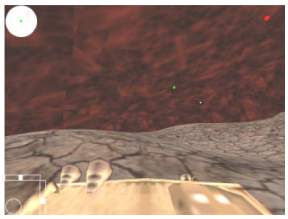
## Camera display modes



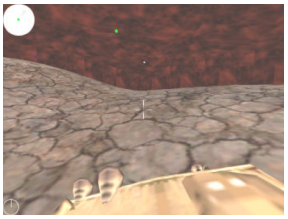

Problems:

- Exocentric views hard to achieve
- A fixed camera on the vehicle may give an illusion of flatness
- The angle of the horizon line gets confused with the roll of the vehicle; the "graveyard spiral" (Roscoe 1999)
- Gravity referenced display with the tilted vehicle's chassi improves situational awarness

## Gravity referenced display



Fixed Camera
(note the roll display in lower left corner)

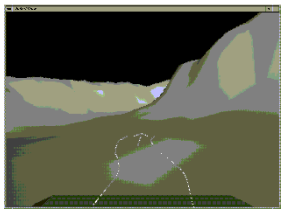


Gravity Referenced Display
(note the indication of roll provided by the tilt of the robot's body)

Pictures: Jijun Wang

61

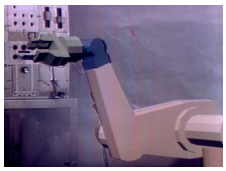
## Predictive Displays

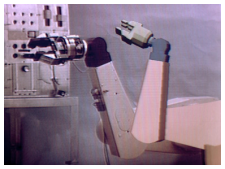


- Predicts 5 seconds ahead by simulation based on user actions and vehicle velocity
- Superimposed information on the display:
  - The length of the lines: Indirect velocity information
  - An arrow describing the vehicle's predicted position and heading
- A representation of the vehicle body

62

## Predictive Displays





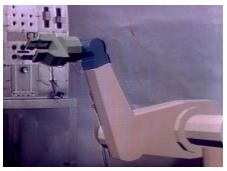
(Kim and Bejczy, 1993)

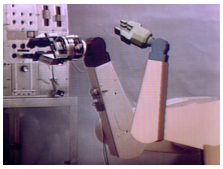
The operator can manipulate a computer graphics simulation of the slave robot.

This simulated robot can be superimposed over the video returning from the remote site

63

## Predictive Displays

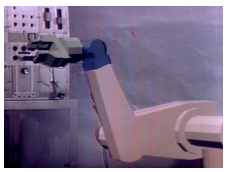


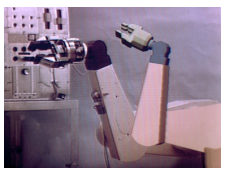


"Time Clutch" : a foot pedal which, when pressed allows the simulated robot to move without the physical robot moving. The operator's inputs are held in

64

## Predictive Displays

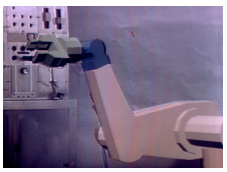


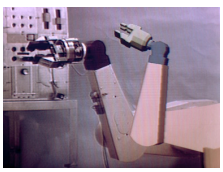


"Position clutch" : disengages the operator's commands entirely from the physical robot so that the operator can fine-tune positioning in the simulator.

65

## Predictive Displays





"Time brake" : emptying out the command memory until the simulated robot "comes back" to the current physical robot state

66

# References

S. Lichiardopol, A Survey on Teleoperation, Technische Universiteit Eindhoven, 2007

M. Endsley, 'Theoretical Underpinning of Situation Awareness: Critical Review (2000)' in Mica R. Endsley and Daniel J. Garland (Eds.) *Situation Awareness Analysis and Measurement.* Lawrence Erlabaum Associates, Mahwah, New Jersey, pp. 3-32, 2000.

Jijun Wang, Michael Lewis, Stephen Hughes, Gravity-Referenced Attitude Display for Teleoperation of Mobile Robots, In PROCEEDINGS of the HUMAN FACTORS AND ERGONOMICS SOCIETY 48th ANNUAL MEETING 2004.