How to Pass the Exam (part 1)

Ian Pratt-Hartmann

Department of Computer Science, Manchester University, UK
email: ian.pratt@manchester.ac.uk

COMP24011: 2022–23

- Basic search
    - Planning and inference tasks as search problems.
    - Depth-first and breadth-first search.
    - Branch-and-bound and A^*.
    - Towers of Hanoi example (from Topic 2).
- R+N Ch. 3.

- Adversarial search
    - Two-player games as adversarial search.
    - Minimax and alpha-beta pruning.
    - Practical refinements.
- R+N Secs. 5.1–5.3.

- Constraint satisfaction
    - Definition of constraint network.
    - Formalizing problems as CSPs.
    - (Directed) arc-, path- and (strong) k-consistency; AC3.
    - (Directed) constraint graphs.
    - Cycle cut-sets and tree-decompositions.
    - Gaschnig backjumping.
- R+N Ch. 6, but not Sec. 6.4.

- Planning
  - Planning rules.
  - Forward chaining (search).
  - Planning graphs (for cost underestimates).
  - Backward chaining
  - Means-ends analysis.
- R+N Secs. 10.1–10.3.

- Logic
    - The situation calculus.
    - Reasoning about the blocks world.
    - Plans as situation-terms.
    - The frame problem and qualification problem.
    - Means-ends analysis.
- R+N Secs. 12.1–12.5 (background).

- Probability
  - Degrees of belief and probability.
  - Updating belief and conditionalization.
  - Synchronic and diachronic Dutch book arguments.
  - Robot localization example. Distributions.
- R+N Ch. 13.

- Bayes' networks
  - Partition (etc.), (conditional) independence.
  - Bayes networks and their motivation.
  - Factorizing probability distributions.
  - Conditioning in Bayes networks.
- R+N Secs. 14.1–14.4.3.

Part II Revision Guide

COMP24011: Introduction to AI
Week 12
Riza Batista-Navarro

Knowledge Representation

Sections 12.1-12.3 of Russell and Norvig

Categories

Level used in **reasoning**

Category information can be used to make predictions about objects

e.g., if an object *f* is a *Fruit*, then it can probably be used in a fruit salad

Inheritance

Properties are **inherited** through membership in categories

E.g., given the following taxonomy:

Food

Fruit

Apples

If all instances of *Food* are *edible,* then every apple is *edible*

Other relations between categories

Disjoint: two or more categories that have no common objects/members

Disjoint({Animals, Vegetables})

Exhaustive decomposition: an object needs to be at least one of the categories

ExhaustiveDecomposition({Males, Females}, Animals)

Partition: a disjoint exhaustive decomposition

Partition({Males, Females}, Animals)

Interval Relations (Allen's Algebra)

Given two intervals *i* and *j:*

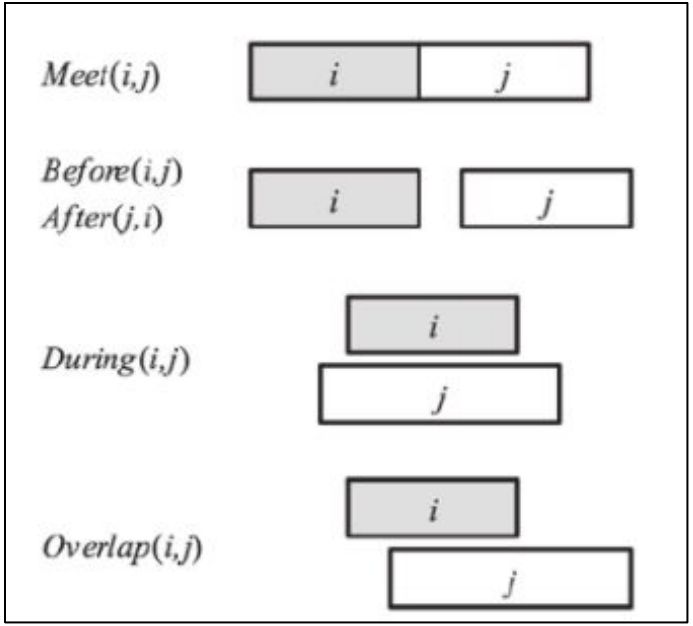
Meet(i, j) ⇔ End(i) = Begin(j)

Before(i, j) ⇔ End(i) < Begin(j)

After(j, i) ⇔ Before(i, j)

During(i, j) ⇔ Begin(j) < Begin(i) < End(i) < End(j)

Overlap(i, j) ⇔ Begin(i) < Begin(j) < End(i) < End(j)



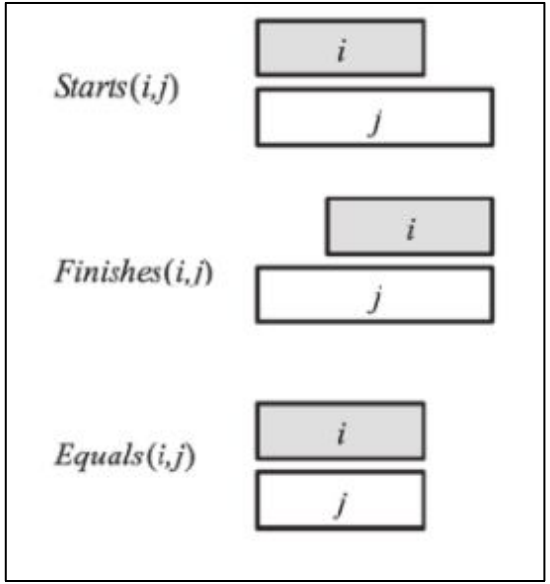
Interval Relations (Allen's Algebra)

Given two intervals *i* and *j:*

Starts(i, j) ⇔ *Begin(i) = Begin(j)*

Finishes(i, j) ⇔ *End(i) = End(j)*

Equals(j, i) ⇔ *Before(i) = Begin(j)* ∧ *End(i) = End(j)*

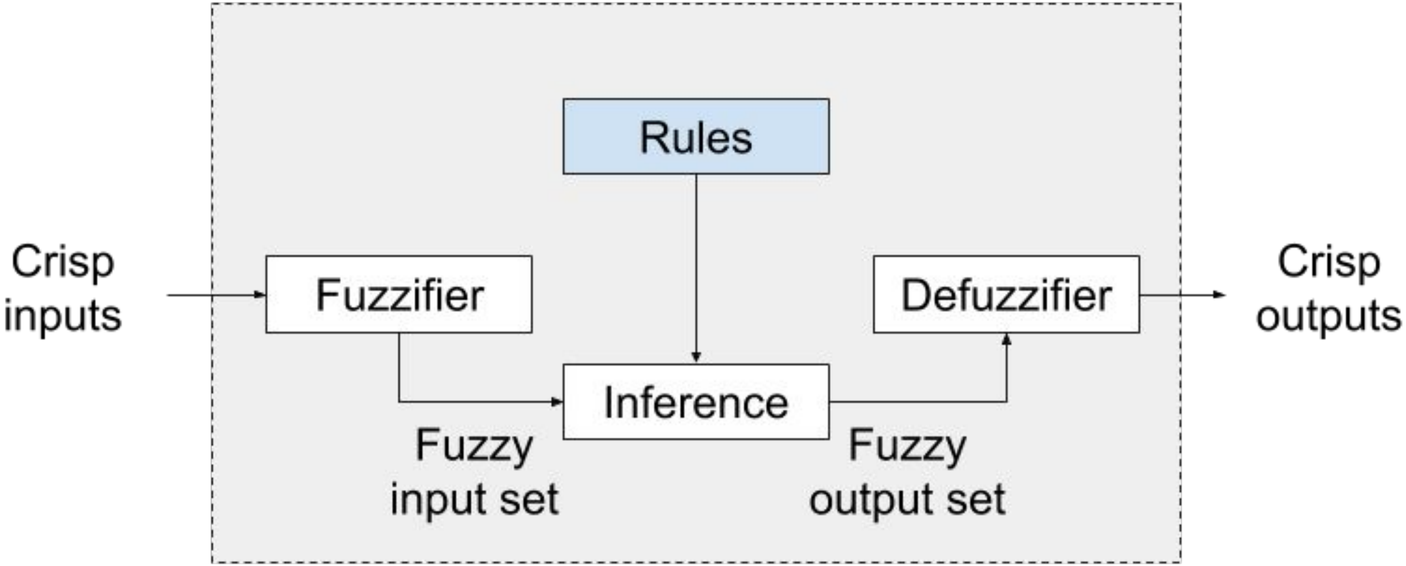


Fuzzy Logic

Section 12.4 of Brachman and Levesque

Fuzzy Control

A method for constructing control systems where **fuzzy rules** are used to map real-valued input (**crisp values**) to output parameters.

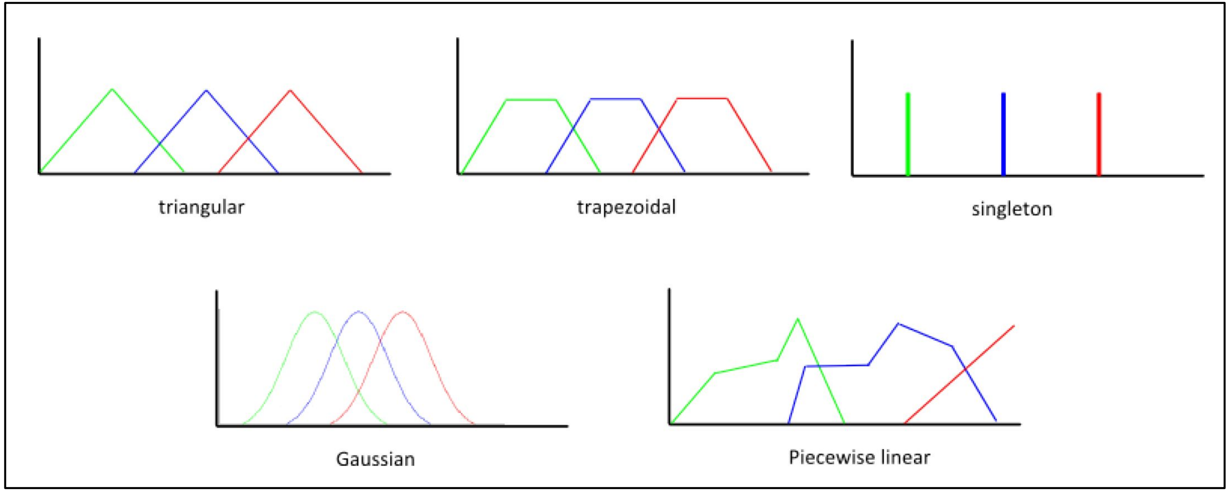


Overall architecture of a fuzzy control system

Initialisation

Membership functions

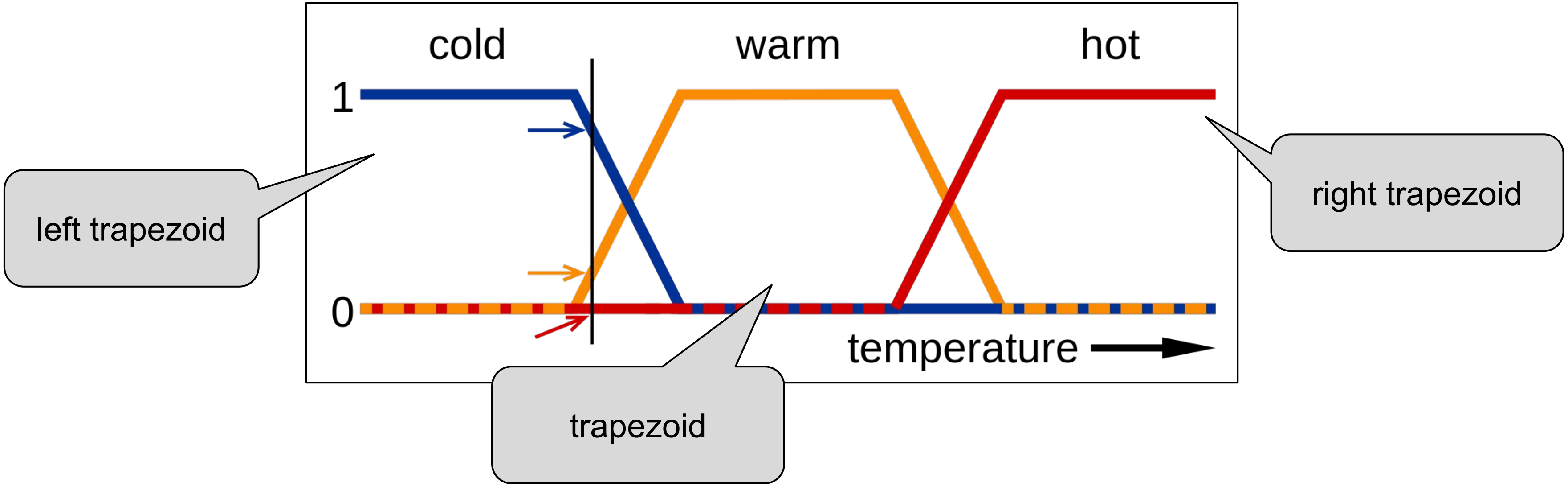
- used in mapping crisp values to linguistic terms (and vice-versa)
- context-dependent; chosen arbitrarily based on user knowledge/experience



Examples of membership functions

Initialisation

Membership functions



Fuzzification

Evaluation steps:

service is poor **OR** *food is rancid*

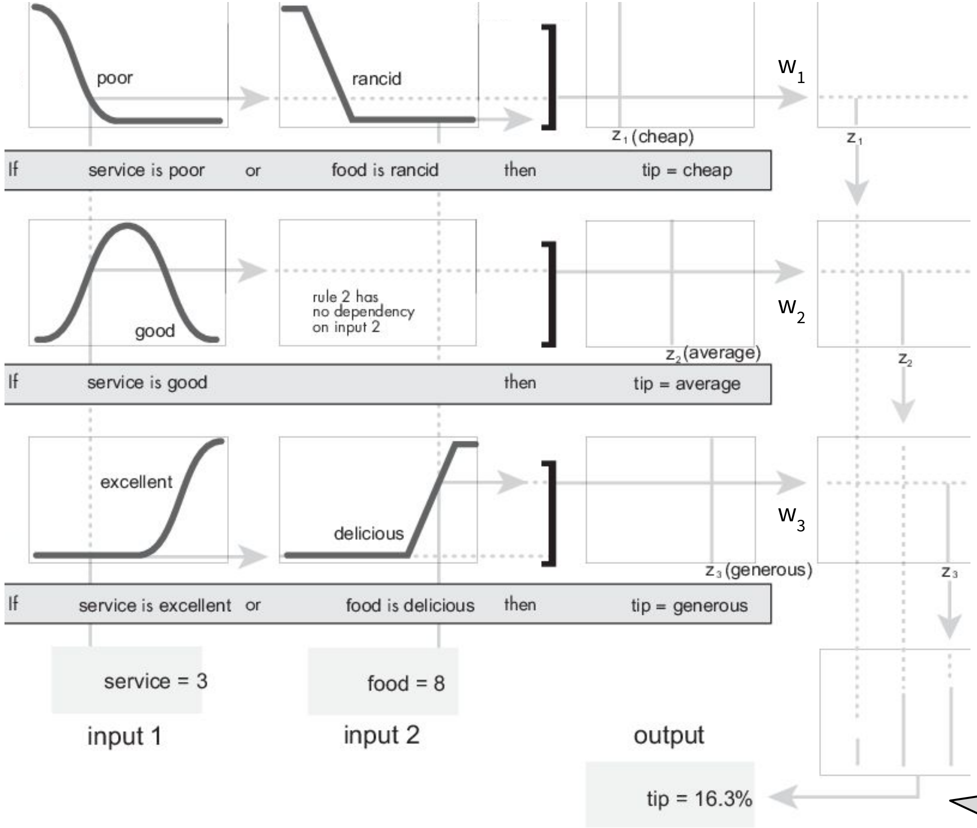
(1) Evaluate each of the two antecedents A and B

Given A [*service is poor*], T(A) is the degree to which a crisp value is *poor*
Given B [*food is rancid*], T(B) is the degree to which a crisp value is *rancid*

(2) Evaluate the overall antecedent

Conjunction (a.k.a. Intersection): $T(A \wedge B) = min(T(A), T(B))$ (AND)
Disjunction (a.k.a.Union): $T(A \vee B) = max(T(A), T(B))$ (**OR**)
Negation (a.k.a. Complement): $T(\neg A) = 1 - T(A)$ (NOT)

Sugeno fuzzy inference



$$\text{weighted average} = \frac{\sum_{i=1}^{N} w_i z_i}{\sum_{i=1}^{N} w_i}$$

Defuzzification (weighted average)

Source: https://uk.mathworks.com/

Natural Language Processing

Sections 22.1-22.5 and Section 23.1 of Russell and Norvig

Text Classification

Standard approach: **Bag of Words**

- represent the input text as feature-value pairs (a **feature vector**)
- E.g., **features**: unigrams, **values**: number of times each unigram appears in the text
- Sparse vector, word order is lost (higher-order *n*-gram word models maintain local order only)

Feature selection: discard frequent bigrams (e.g., *"of the"*) appearing in both classes

Supervised learning: support vector machines, decision trees,
logistic regression

Information Retrieval (IR)

Earliest approach: **Boolean keyword model**

- each word in the corpus is a Boolean feature
- 1 for document *d* if the word appears in *d;* 0 otherwise
- query is a Boolean expression over features, e.g., *[information AND retrieval]*
- drawback: difficult to present result set in order of relevance

Information Retrieval (IR)

Approach based on a scoring function: **BM25**

- takes a query and a document, outputs a numeric score
- the score is a linear weighted combination of scores for each word in the query
- given a document d_j and a query consisting of a sequence of words $q_{1:N}$

$$BM25(d_j, q_{1:N}) = \sum_{i=1}^{N} IDF(q_i) \cdot \frac{TF(q_i, d_j) \cdot (k+1)}{TF(q_i, d_j) + k \cdot (1 - b + b \cdot \frac{|d_j|}{L})}$$

Information Retrieval (IR)

$$BM25(d_j, q_{1:N}) = \sum_{i=1}^{N} IDF(q_i) \cdot \frac{TF(q_i, d_j) \cdot (k+1)}{TF(q_i, d_j) + k \cdot (1 - b + b \cdot \frac{|d_j|}{L})}$$

three factors in BM25:

- term frequency *TF*: how frequent a word appears in a document
- inverse document frequency *IDF*: inverse of document frequency *DF* (number of documents containing the word)
- length of the document $|d_j|$

Information Retrieval (IR)

Formula to be provided but without variable definitions

$$BM25(d_j, q_{1:N}) = \sum_{i=1}^{N} IDF(q_i) \cdot \frac{TF(q_i, d_j) \cdot (k+1)}{TF(q_i, d_j) + k \cdot (1 - b + b \cdot \frac{|d_j|}{L})}$$

where:

N = number of documents in the corpus

$|d_j|$ = length of the document

L = average document length in the corpus

k = parameter typically 2.0

b = parameter typically 0.75

Information Retrieval (IR)

Inverse document frequency: measures importance of a word

$$IDF(q_i) = \log \frac{N - DF(q_i) + 0.5}{DF(q_i) + 0.5}$$

Information Retrieval (IR)

Evaluation:

	In result set	Not in result set
Relevant	30	20
Not relevant	10	40

Precision: proportion of documents in the result set that are relevant

= 30 / (30 + 10) = 0.75 **False positive rate**: 1-0.75 = 0.25

Recall: proportion of all relevant documents, that are in the result set

= 30 / (30 + 20) = 0.60 **False negative rate**: 1-0.60 = 0.40

F1 score: harmonic mean of precision and recall

= 2PR/(P+R) = 0.9/1.35 = 0.67

Information Retrieval (IR)

Refinements:

- case folding (e.g., "COUCH" --> "couch")
- stemming (e.g., "couches" --> "couch")
  but can hurt precision, e.g., "stocking" --> "stock"
- recognising synonyms (e.g., "couch" and "sofa")
  but can also hurt precision, e.g., "couch" in *[Tim Couch]* does not refer to sofas

Information Retrieval (IR)

PageRank algorithm: if the query is *[IBM]*, it is desirable to return IBM's home page as the top result, even if another document mentions "IBM" more frequently

in-links: links to a page

out-links: links from a page

(Recursive) Definition of PageRank for a page *p*:

$$\boxed{PR(p) = \frac{1-d}{N} + d\sum_i \frac{PR(in_i)}{C(in_i)}}$$

where: in_i = in-links to *p*, $C(in_i)$ = count of out-links on in_i

d = damping factor (probability a web surfer clicks on a link)

Information Extraction (IE)

Similar to skimming a piece of text to find **instances** (or entities) of a particular type of object, or **relationships** between them

Approaches:

- Finite state automata
- Probabilistic models
- Conditional random fields

Finite state automata

Pattern or template: defined using a finite state automaton, i.e., a **regular expression**

`[0-9]`	matches any digit from 0 to 9
`[0-9]+`	matches one or more digits
`[.][0-9][0-9]`	matches a period followed by two digits
`([.][0-9][0-9])?`	matches a period followed by two digits, or nothing
`[$][0-9]+([.][0-9][0-9])?`	matches $249.99 or $1.23 or $1000000 or …

Can extract individual attributes

For relations, **cascaded finite-state transducers** can be used

Probabilistic Context-free Grammar (PCFG)

Grammar: collection of rules used to define a language as a set of allowable strings of words

Context-free grammars (CFGs): a set of production rules where each one is of the form:

$A \rightarrow \alpha$

where: A is a single **non-terminal** symbol

α is a string of **terminal** (actual words) or **non-terminal** symbols

Each rule allows rewriting the non-terminal as the RHS in any context

Probabilistic Context-free Grammar (PCFG)

Probabilistic CFG: the grammar assigns a probability to every string

$$\begin{array}{lcl} VP & \rightarrow & Verb\ [0.70] \\ & | & VP\ NP\ [0.30] \end{array}$$

where: VP stands for verb phrase

NP stands for noun phrase

A toy language ε_0

Lexical categories: some categories are **closed** classes, others **open**

Noun	$\rightarrow$	**stench** [0.05] | **breeze** [0.10] | **wumpus** [0.15] | **pits** [0.05] | ...
Verb	$\rightarrow$	**is** [0.10] | **feel** [0.10] | **smells** [0.10] | **stinks** [0.05] | ...
Adjective	$\rightarrow$	**right** [0.10] | **dead** [0.05] | **smelly** [0.02] | **breezy** [0.02] ...
Adverb	$\rightarrow$	**here** [0.05] | **ahead** [0.05] | **nearby** [0.02] | ...
Pronoun	$\rightarrow$	**me** [0.10] | **you** [0.03] | **I** [0.10] | **it** [0.10] | ...
RelPro	$\rightarrow$	**that** [0.40] | **which** [0.15] | **who** [0.20] | **whom** [0.02] $\vee$ ...
Name	$\rightarrow$	**John** [0.01] | **Mary** [0.01] | **Boston** [0.01] | ...
Article	$\rightarrow$	**the** [0.40] | **a** [0.30] | **an** [0.10] | **every** [0.05] | ...
Prep	$\rightarrow$	**to** [0.20] | **in** [0.10] | **on** [0.05] | **near** [0.10] | ...
Conj	$\rightarrow$	**and** [0.50] | **or** [0.10] | **but** [0.20] | **yet** [0.02] $\vee$ ...
Digit	$\rightarrow$	**0** [0.20] | **1** [0.20] | **2** [0.20] | **3** [0.20] | **4** [0.20] | ...

Sum of probabilities per category = 1.0

A toy language ε_0

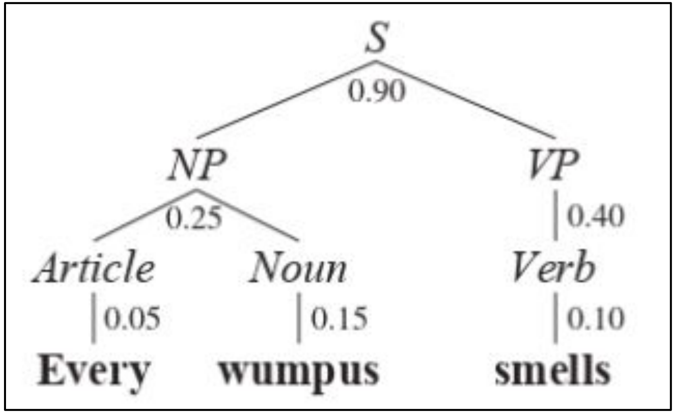
Syntactic categories

S	$\rightarrow$	$NP\ VP$	[0.90]	I + feel a breeze
	$\mid$	$S\ Conj\ S$	[0.10]	I feel a breeze + and + It stinks
NP	$\rightarrow$	$Pronoun$	[0.30]	I
	$\mid$	$Name$	[0.10]	John
	$\mid$	$Noun$	[0.10]	pits
	$\mid$	$Article\ Noun$	[0.25]	the + wumpus
	$\mid$	$Article\ Adjs\ Noun$	[0.05]	the + smelly dead + wumpus
	$\mid$	$Digit\ Digit$	[0.05]	3 4
	$\mid$	$NP\ PP$	[0.10]	the wumpus + in 1 3
	$\mid$	$NP\ RelClause$	[0.05]	the wumpus + that is smelly
VP	$\rightarrow$	$Verb$	[0.40]	stinks
	$\mid$	$VP\ NP$	[0.35]	feel + a breeze
	$\mid$	$VP\ Adjective$	[0.05]	smells + dead
	$\mid$	$VP\ PP$	[0.10]	is + in 1 3
	$\mid$	$VP\ Adverb$	[0.10]	go + ahead
$Adjs$	$\rightarrow$	$Adjective$	[0.80]	smelly
	$\mid$	$Adjective\ Adjs$	[0.20]	smelly + dead
PP	$\rightarrow$	$Prep\ NP$	[1.00]	to + the east
$RelClause$	$\rightarrow$	$RelPro\ VP$	[1.00]	that + is smelly

A toy language ε_0

Phrase structure

For *"Every wumpus smells"*



Probability of the tree as a whole:

0.90 x 0.25 x 0.05 x 0.15 x 0.40 x 0.10

Computer Vision

Section 4.1 of Szeliski

Feature Matching

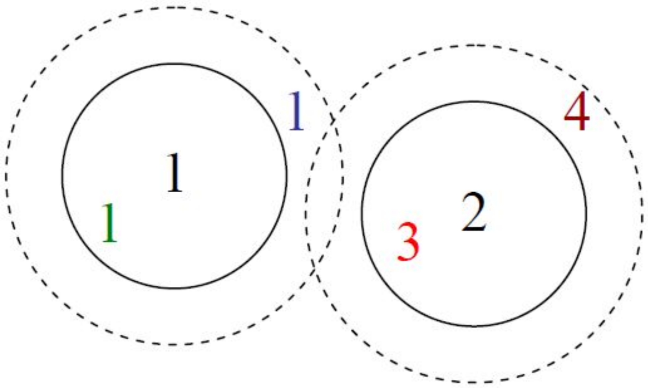
Searching for likely matches (matching descriptors) in a number of images/frames

Matching strategies

- **Euclidean distance** (setting a threshold)
- **Nearest neighbour**
- **Nearest neighbour distance ratio (NNDR) matching**

Feature Matching: Euclidean Distance

Set a threshold (maximum distance); return all matches from other images within this threshold



	t1	t2
True positives (TP)	1	1, 1
False positives (FPs)	3	3, 4
False negatives (FNs)	1	

Feature Matching: Euclidean Distance

Confusion matrix

	True matches	True non-matches
Predicted matches	18 (TP)	4 (FP)
Predicted non-matches	2 (FN)	76 (TN)

Precision = TP/(TP+FP) = 18/(18+4) = 18/22 = 0.82

Recall = TP/(TP+FN) = 18/(18+2) = 18/20 = 0.90

Positive predictive value (PPV) = same as **Precision** = 0.82

Feature Matching: Nearest Neighbour (NN)

Simply take the nearest neighbour(s)

Threshold still useful to reduce FPs (since some features might not have any matches at all)

Nearest Neighbour Distance Ratio (NNDR)

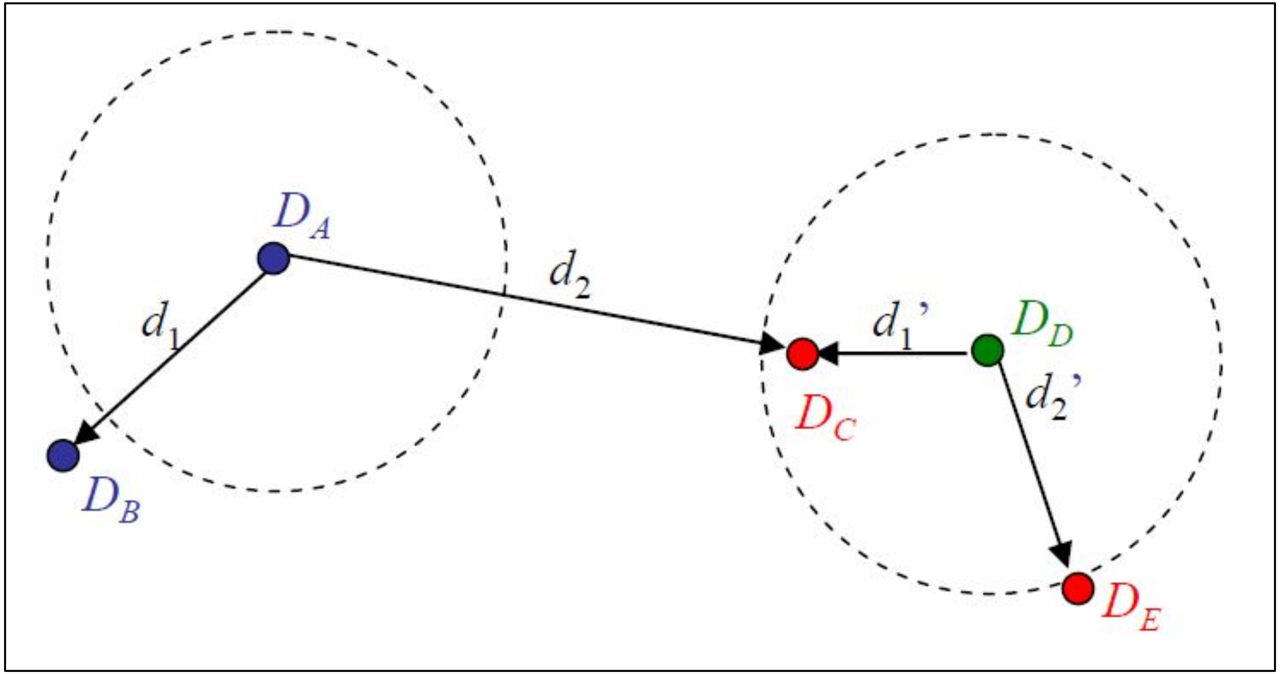
d_1/d_2 where d_1 is the distance between the target descriptor and the NN

d_2 is the distance between the target descriptor and the 2nd NN

a low ratio indicates a good match; a high ratio indicates ambiguity

Feature Matching:

Fixed threshold vs **NN** vv **NNDR**



Robotics

Sections 25.1-25.3 of Russell and Norvig

Robot Hardware: Effectors

The means by which robots **move** and change the **shape** of their bodies

Degree of freedom (DoF): an independent direction in which a robot (or its effector) can move

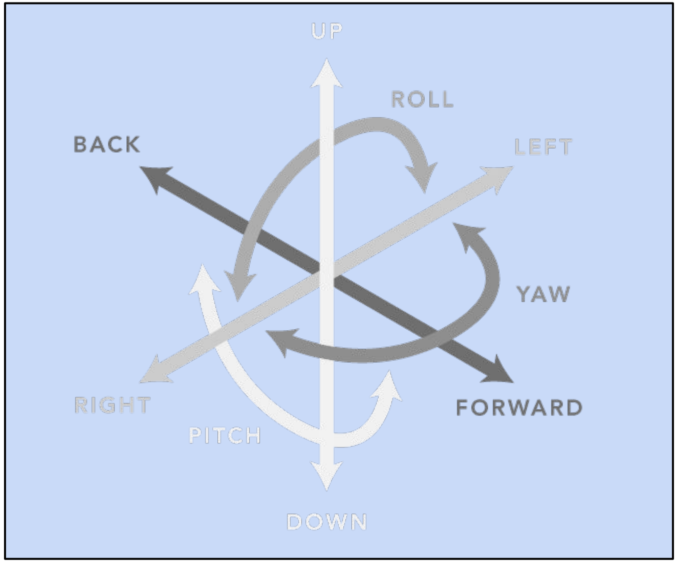
6 DoF (a.k.a. kinematic state) : x, y, z (location)
yaw, pitch, roll (angular direction a.k.a. orientation)

dynamic state: 6 DoF plus six dimensions for the rate of change (**velocity**) of each dimension

Robot Hardware: Effectors

6 DoF : **x** (left or right), **y** (forwards or backwards), **z** (up or down)

yaw (turning from left to right or vice-versa), **pitch** (tilting forwards or backwards), **roll** (tilting side to side)

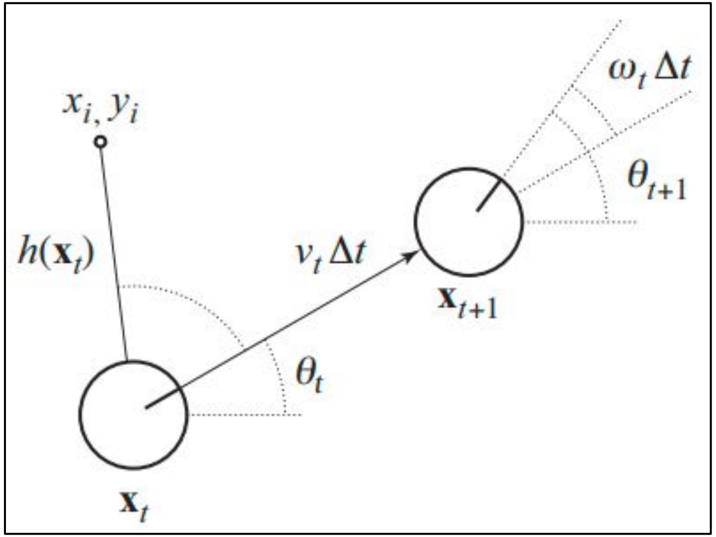


Localisation and Mapping

Localisation: finding out where things are (including the robot itself)

Robot pose: defined by Cartesian coordinates *x* and *y*, and *heading* (orientation)

$$\boldsymbol{X}_t = (x_t, y_t, \theta_t)^T$$



Localisation and Mapping

Action: specified as two velocities

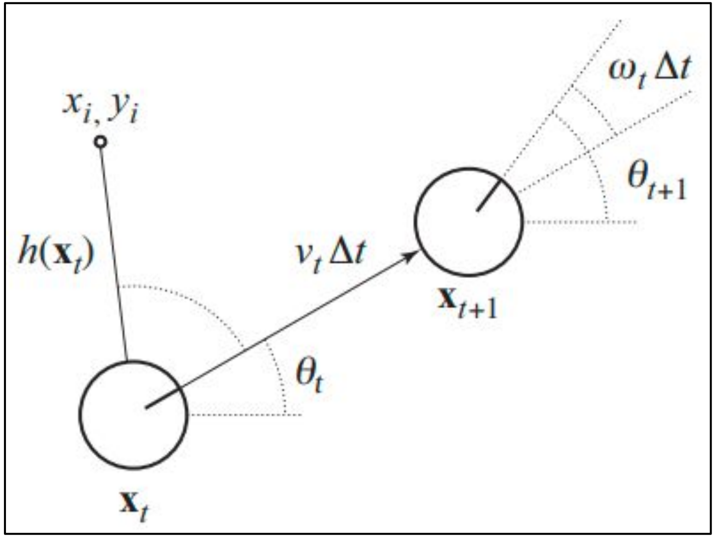
Translational v_t

Rotational ω_t

New state $\boldsymbol{x}_{t+1}$ can be obtained as:

update in position $v_t \Delta t$

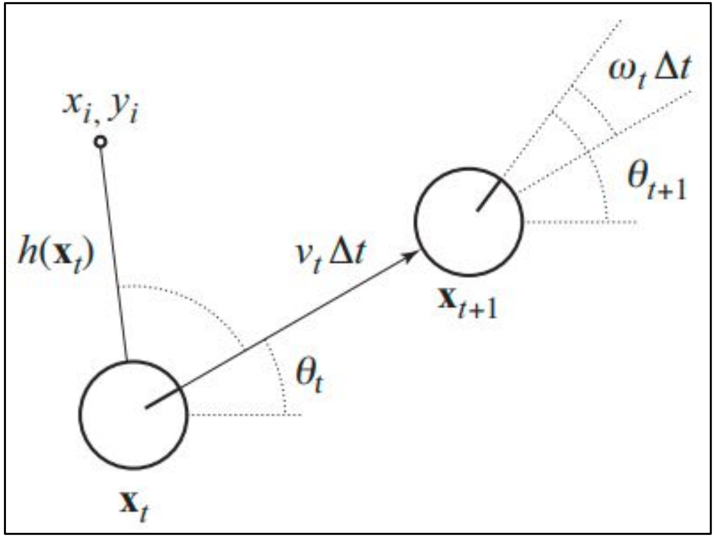
update in orientation $\omega_t \Delta t$



Localisation and Mapping

Landmark as observation

- stable, recognisable features of the environment
- measured in terms of **range** (relative distance) and **bearing** (relative orientation)



$$\hat{\mathbf{z}}_t = h(\mathbf{x}_t) = \begin{pmatrix} \sqrt{(x_t - x_i)^2 + (y_t - y_i)^2} \\ \arctan \frac{y_i - y_t}{x_i - x_t} - \theta_t \end{pmatrix}$$

General tips/notes

- Calculations
  - You can use a calculator
  - When calculating the log(x) use the common logarithm: base 10
  - How to round off your final answer: there will be a specific instruction alongside each numeric calculation question
- If the question is asking for a short answer, and your answer consists of multiple items, put one item per line
- Some answers might require case sensitivity

Thank you!