Artificial Intelligence at Purdue

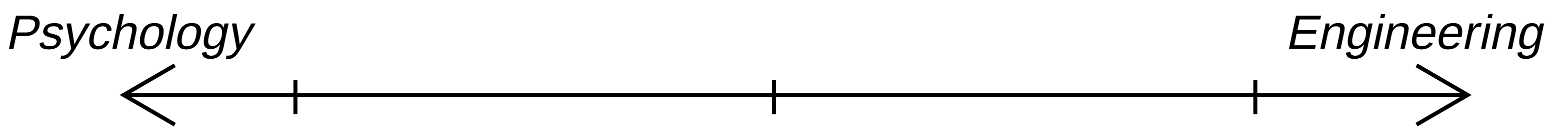
Robert Givan

Electrical & Computer Engineering

Purdue University

What is Artificial Intelligence?

Many definitions vie for attention:

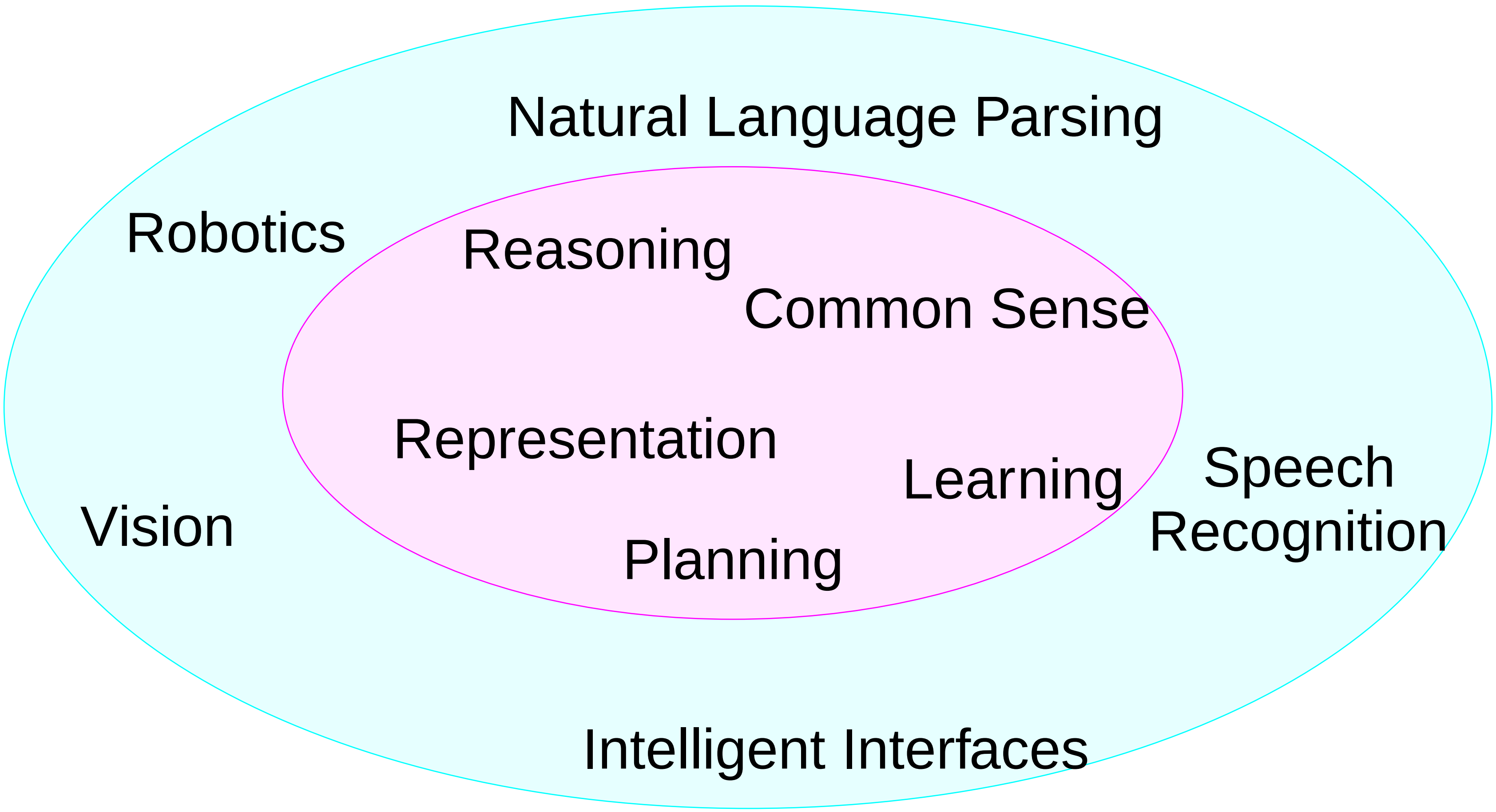


Making computers do things *the way people do them...*

Making computers do things *that people do... (by any mechanism)*

Making computers do things *that they can't currently do (that seem intelligent)*

What are the subfields of AI?



Some AI History

1950's: AI will be really easy

1960's: AI will be pretty easy

1970's: AI is really hard

Focus on Cognitive AI

1980's: AI is really hard, but it sells really well!

1990's: We can solve small pieces of AI

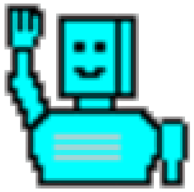
We can show specific progress.

Focus on Interactive AI

Modern AI is very different from traditional AI.

Traditional AI— A Caricature

Three steps to HAL 2000:

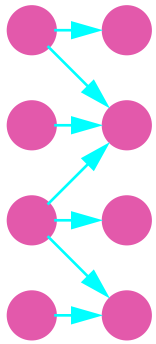


1. Write down what you know in a formal logic
2. Code up a general purpose theorem prover
3. Have a conversation with it:
   a. Translate your comments to theorems
   b. Translate your questions to logical queries
   c. Translate its proofs back to natural language as answers to the queries.

Other early approaches were equally naive...

Modern AI

- Traditional core AI tried for *simple, general-purpose solutions.*
- Modern core AI looks for smaller pieces that can be solved with new algorithms/representations.
- Typical tools:
  - Compact representations
  - Algorithms designed for compact representations
  - Representation choice to facilitate these algorithms



Typically, resulting solutions are near useful application.

Faculty doing AI work at Purdue

- (Charlie Bouman............................ image processing)
- (Carla Brodley......... data mining for computer security)
- Bob Givan..... machine learning, planning, & reasoning
- Mary Harper ....speech/language/gesture recognition
- Avi Kak.......machine sensory intelligence (robot vision)
- Jeff Siskind ....needs a slide all to himself!!
- Phil Swain.... AI methods to enhance teaching/learning

Jeff Siskind

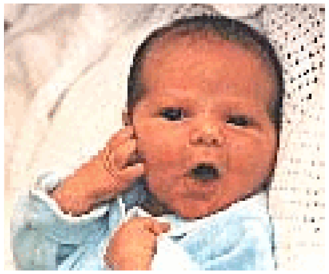
- Computational models of child language acquisition
- Grounding natural language semantics in vision
- Visual event perception
- Image segmentation
- Parsing images with probabilistic context-free grammars

Work outside AI:

- Whole program optimization
- Programming environments for worldwide distributed shared source-code repositories

Acquiring Word Meanings

Imagine an "infant program"



Inputs:

- processed visual input (perhaps with objects noted)
- processed linguistic input (perhaps w/words noted)

Output:

- Lexicon of word meanings (& not just simple nouns)

How does an infant do it??

My Own Work in AI

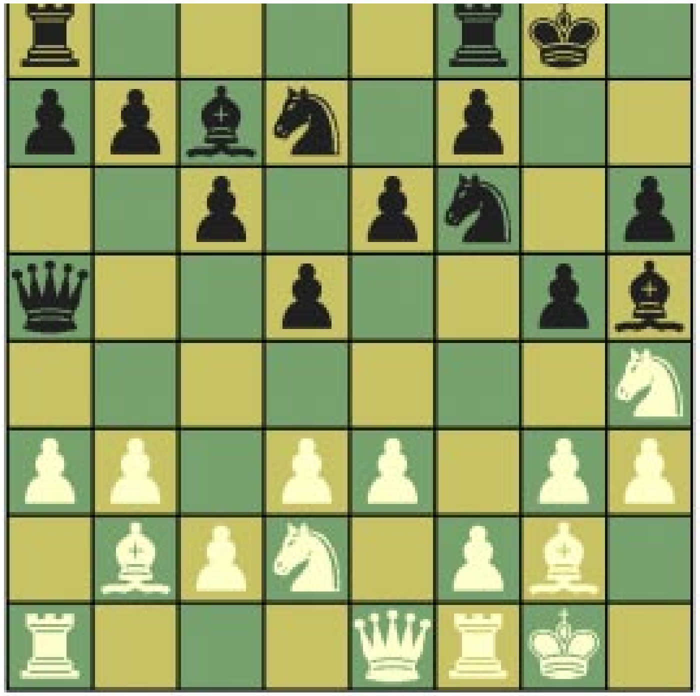
- Reasoning: -quickly inferring the obvious
  -e.g. smarter compilers
- Planning: -using reasoning and learning to plan
  -compact problem representation
  -handling uncertainty
- Learning -planning by learning from experience
  -learning for branch prediction
  - learning word meanings from visual input
- Representation: -class-based logic

I am interested in talking to students with interests in any of these AI areas.

Deciding What is (Obviously) True

Obvious: Easily discovered, seen, or understood; readily perceived by the eye or the intellect; plain; evident; apparent;

Webster's Revised Unabridged



Does your Compiler Understand your Program?

```
sorted_list* sort (numlist* lst) {
   if (lst==NULL)
       return(NULL)
   else
       return( insert( lst->num,
                       sort(lst->next) ));
}
```

We'd like it to be *obvious* that this program sorts a list.

- *i.e.,* it returns a list with exactly the same elements that were in LST, but in sorted order.

No compiler today can verify this.

Most Programs *Obviously* Terminate

Unless there is an error:

```
int factorial (int n) {
   if  (n == 0)
       return(1)
   else
       return( n * factorial (1+ n))
}
```

We'd like a compiler that can warn us when our programs don't *obviously terminate*.

Planning — Deciding What to do Next

Input:

- Your knowledge of the world
- Your knowledge of the likely effects of your actions
- Some goal or utility function

Output:

- A “plan”: what actions should you take?

How to find the plan? How to represent the plan?

How to even represent the problem?