

Intro to AI for K-12 Educators

David S. Touretzky
Carnegie Mellon University

RockCS — June 5, 2021.



RockCS
ROCKY MOUNTAIN COMPUTER SCIENCE
CONFERENCE FOR P-12 EDUCATORS



What is AI?

Artificial Intelligence is a
branch of computer science
concerned with techniques that
allow **computers** to do things
that, when people do them, are
considered evidence of
intelligence.

We Use AI-Powered Technologies Every Day

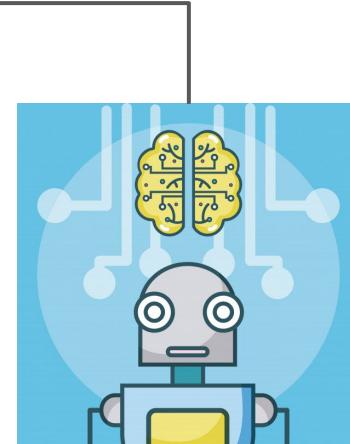
- Speech recognition
 - Home assistants: “Alexa, please dim the lights.”
 - Automated video subtitles; automated transcription services (otter.ai, sonix.ai)
- Computer vision
 - Use your face to unlock your iPhone; play with Snapchat and Tiktok filters
 - Self-driving cars: Tesla autopilot; automated lane departure warnings; emergency braking
- Language understanding
 - Google search: “What does an alligator weigh?” “What’s the second largest city in Honduras?”
 - Google Translate: automatic translation among any of 100+ languages; translate menus
- Recommender systems
 - Netflix→movies; Amazon→products; Facebook→news stories; Google→ads
- Robotics
 - Roombas clean up; Kiva robots automate warehouses; drones follow you

AI Poses Challenges for Society

- Many types of work will become more highly automated.
 - Some people will **lose their jobs** -- including white collar jobs.
 - People will find that their **jobs have changed**; new skills are needed.
 - **New jobs will be created** around AI, robotics, and machine learning.
- Automated decision making systems can be opaque and biased.
- AI is powering the surveillance state:
 - Ubiquitous face recognition: loss of privacy
 - Automated monitoring of phone calls and social media
- Deep fakes: seeing is no longer believing; hard to know what to trust

AI is a Branch of Computer Science

- Algorithms and complexity theory
- **Artificial intelligence** ←
- Computer systems (i.e., hardware)
- Databases
- Graphics
- Human-computer interaction
- Networking
- Operating Systems
- Programming Languages
- Security
- Software Engineering



AI's Relationships with Other Disciplines



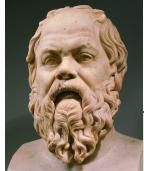
Statistics

Bayesian inference,
regression



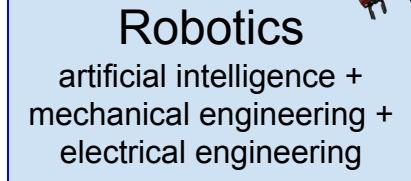
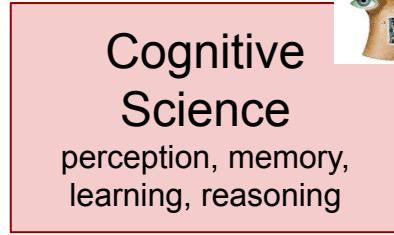
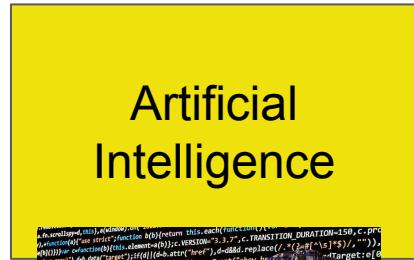
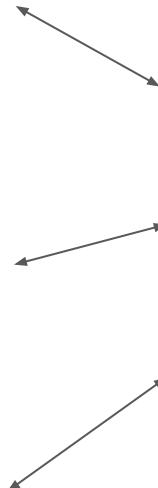
Operations Research

planning, optimization



Philosophy

logic, ontology, theory of
mind, consciousness



Specialties Within AI

- Computer vision
- Game playing (chess, go, poker, etc.)
- Knowledge representation
- **Machine learning**
- Machine translation
- Natural language understanding
- Neural networks
- Optimization
- Planning
- Search
- Speech recognition
- Theorem proving / automated reasoning

These are topics that would be covered in a university-level AI course.

Each topic might itself be a course in a graduate-level AI program.

Weak vs. Strong AI (or Narrow vs. Broad AI)

Weak AI is what we have now:

- Specialized algorithms for solving hard but narrowly-defined problems.
- When IBM's Deep Blue beat chess grandmaster Garry Kasparov in 1996-97, it had no idea who Kasparov was, or what a grandmaster was, or even what physical chess pieces look like.
- Google cannot tell you if an alligator weighs more than an ostrich.

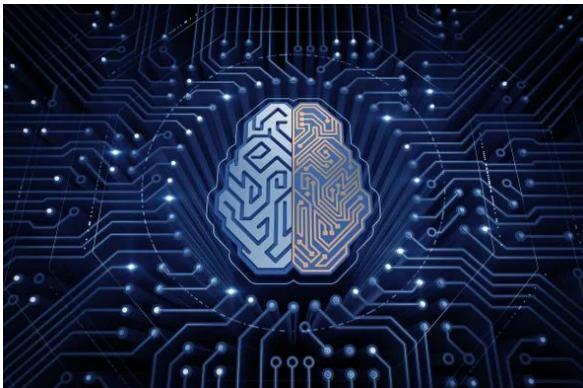
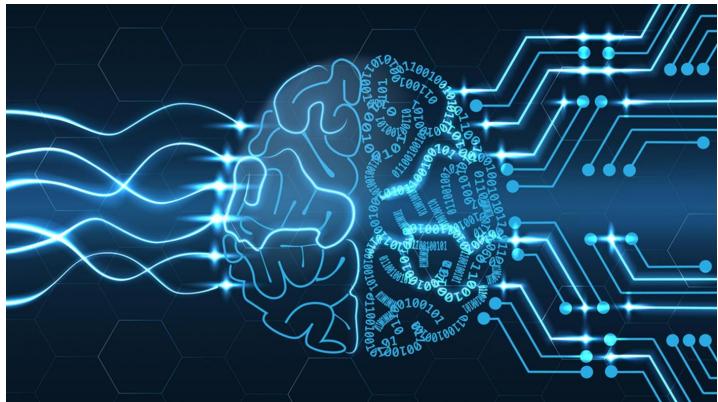
Strong AI or AGI (Artificial General Intelligence) is what we hope to achieve:

- General reasoning abilities with human-like breadth and flexibility
- Self-awareness, consciousness
- This is what is needed to create Commander Data or C3PO or Terminator
- Sadly, we have no idea how to achieve this

Hype vs. Reality

“Google, What Does AI Look Like?”

AI is Blue and Has Brains and Wires and Cyborgs



What AI Really Looks Like: Algorithms + Data Structures

```

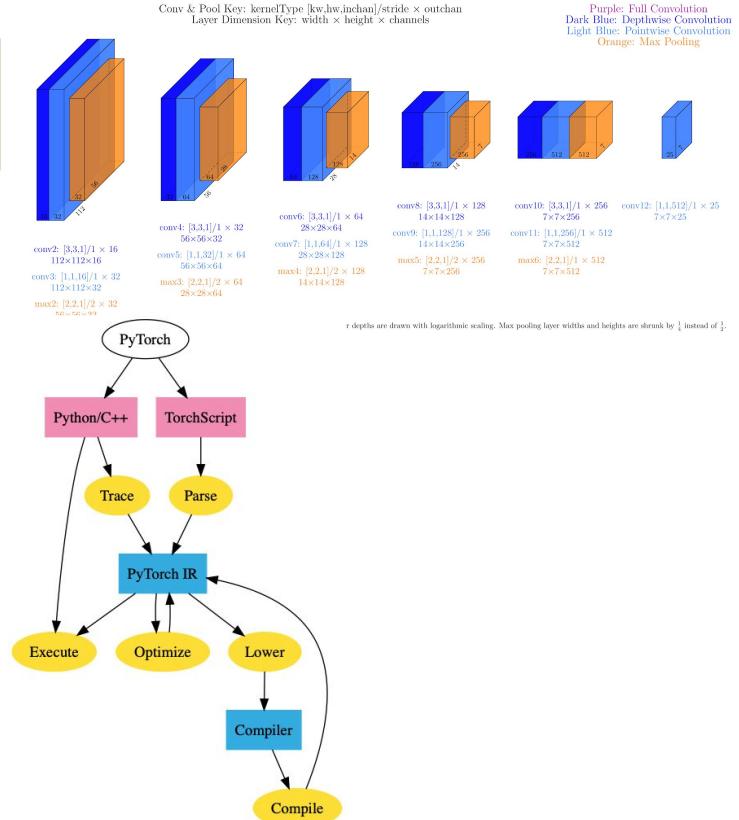
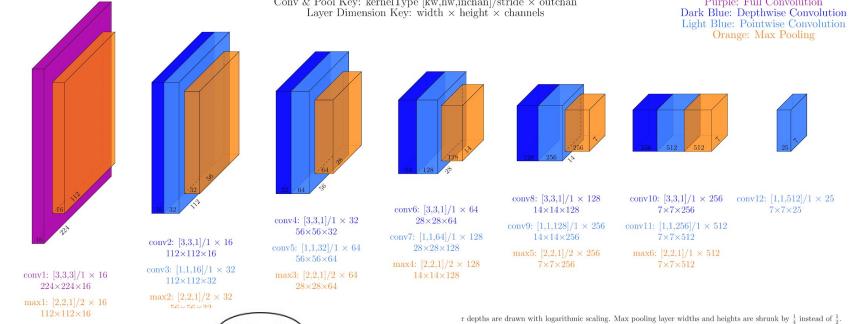
5 import tensorflow as tf
6
7 x = tf.placeholder(tf.float32)
8 W = tf.Variable(0 , dtype = tf.float32)
9 b = tf.Variable(0 , dtype = tf.float32)
10 y_ = tf.placeholder(tf.float32)
11
12 y = W * x + b      calculate the lost
13
14 lost = tf.reduce_mean(tf.square(y_ - y))
15
16 optimizer = tf.train.GradientDescentOptimizer(0.01)
17 train_step = optimizer.minimize(lost)
18
19 sess = tf.Session()
20 init = tf.global_variables_initializer()
21 sess.run(init)
22
23 tf.summary.scalar('lost' , lost)
24 merged = tf.summary.merge_all()
25 writer = tf.summary.FileWriter("./logs" , graph=tf.get_default_graph())
26
27 steps = 1000
28 for i in range(steps):
29     xs = 1
30     ys = 3
31     feed = { x: xs , y_: ys }
32     sess.run(train_step , feed_dict=feed)
33     xs = 2
34     ys = 5
35     feed = { x: xs , y_: ys }
36
37     , rs = sess.run([train_step , merged] , feed_dict=feed)
38     writer.add_summary(rs , i)
39
40 print("W: %f" % sess.run(W))
41 print("b: %f" % sess.run(b))

```

the function of the code is
linear recurrence
 $y = 2x + 1$

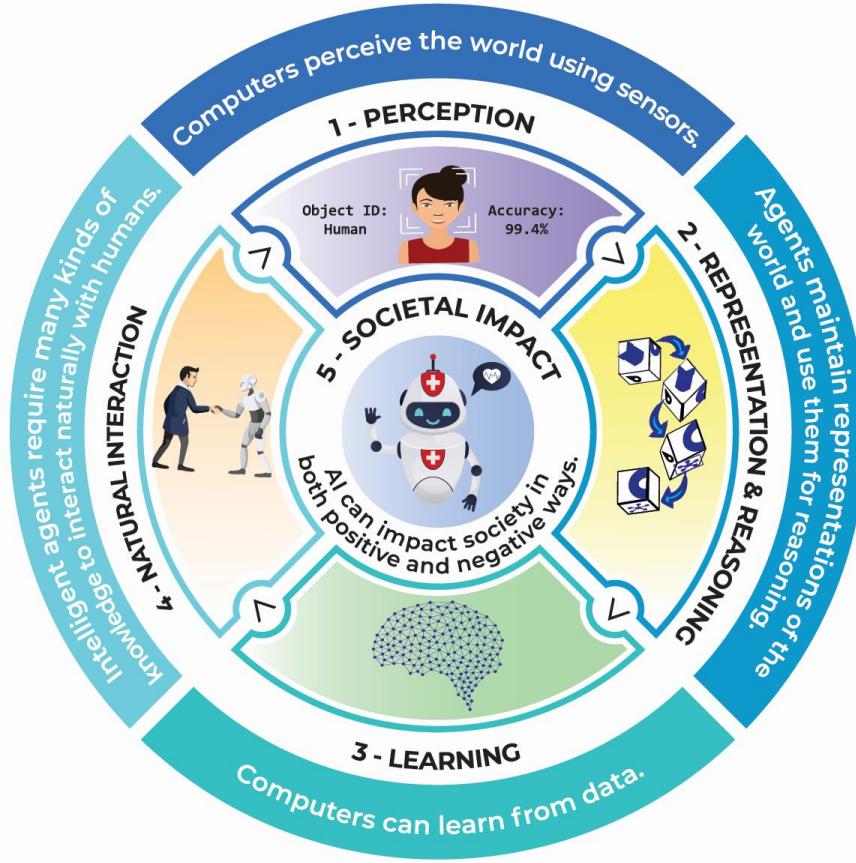
record the lost

update



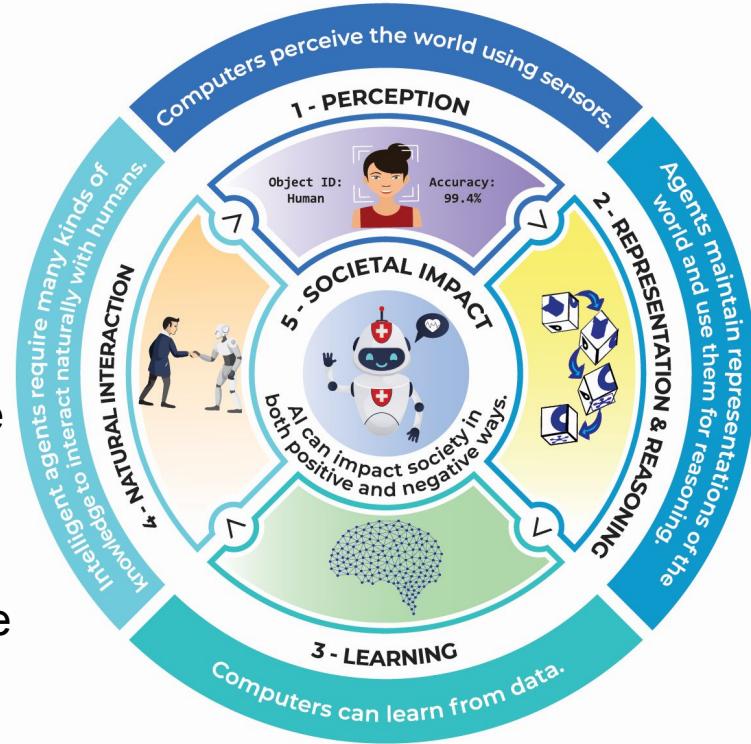
Five Big Ideas in AI

- Organizing framework for the K-12 guidelines, modeled after CSTA Computing Standards.
- 5 big ideas are enough to cover the richness of the field, but small enough to be manageable by teachers.
- CSTA experience shows 5 is a good number.
- Not necessarily the way AI practitioners view their field, but appropriate for the needs of the K-12 audience.



Five Big Ideas in AI

- 1. Perception:** Computers perceive the world using sensors.
- 2. Representation and reasoning:** Agents maintain representations of the world and use them for reasoning.
- 3. Learning:** Computers can learn from data.
- 4. Natural interaction:** Intelligent agents require many kinds of information to interact naturally with humans.
- 5. Societal impact:** AI can impact society in both positive and negative ways.

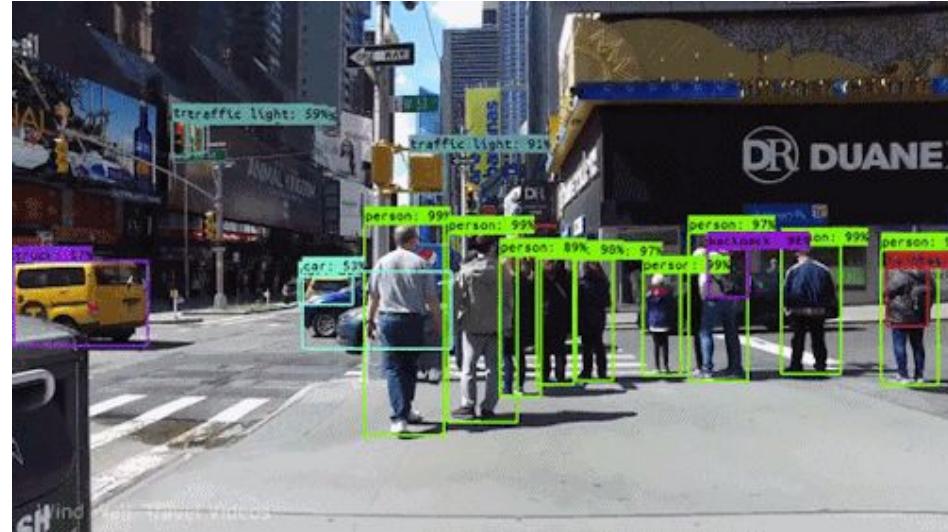


Big Idea #1: Perception

Computers perceive the world using sensors.

Perception is the extraction of *meaning* from sensory signals using knowledge.

- Human senses vs. computer sensors
- Types of perception: vision, speech recognition etc.
- How perception works: algorithms



Example Guidelines

- Identify sensors on computers, robots, and intelligent appliances.
- Explain how sensor limitations affect computer perception.
- Explain that perception systems may draw on multiple algorithms as well as multiple sensors.
- Build an application using multiple sensors and types of perception (possibly with Scratch plugins, or Calypso).

Are Supermarket Doors Intelligent?

This is what you get when your automatic doors have sensing but not perception.



Big Idea #2: Representation and Reasoning

Agents maintain representations of the world, and use them for reasoning.

- Types of representations
- Families of algorithms and the work they do
- Representation supports reasoning: algorithms operate on representations

Example Guidelines

- Create/design a representation of an (animal) classification system using a tree structure.
- Draw a search tree for tic-tac-toe
- Describe how AI representations support reasoning to answer questions
- Describe the differences between types of search algorithms



Big Idea #3: Learning

Computers can learn from data.

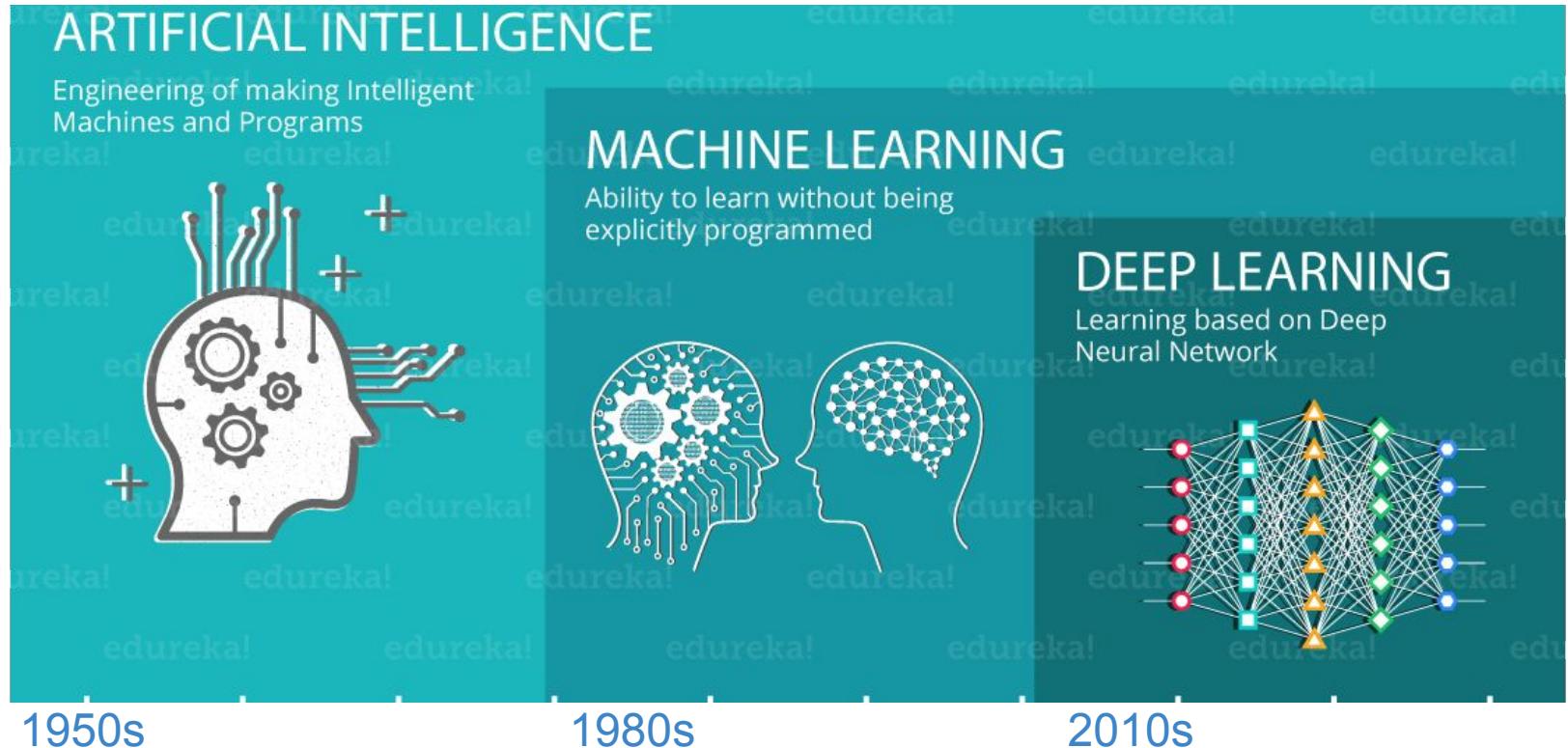
- Nature of learning
- Fundamentals of neural networks
- Data sets



Example Guidelines

- Modify an interactive machine learning project by training its model..
- Describe how algorithms and machine learning can exhibit biases.
- Identify bias in a training data set and extend the training set to address the bias
- Train a neural net (1-3 layers) using *TensorFlow Playground*
- Trace and experiment with a simple ML algorithm

AI and Machine Learning



Big Idea #4: Natural Interaction

Intelligent agents require many kinds of knowledge to interact naturally with humans.

- Natural language understanding
- Common sense reasoning
- Affective computing & interaction (e.g. with robots, or speech agents)
- Consciousness and philosophy of mind

Example Guidelines

- Recognize and label facial expressions into appropriate emotions (happiness, sadness, anger) and explain why they are labeled the way they are
- Experiment with software that recognizes emotions in facial expressions
- Construct a simple chatbot
- Describe some tasks where AI outperforms humans, and tasks where it does not
- Explain and give examples of how language can be ambiguous
- Reason about the nature of intelligence, and identify approaches to determining whether an agent is or is not intelligent.



Big Idea #5: Societal Impact (1 of 3)

“Artificial Intelligence can impact society in both positive and negative ways.”

- **Ethics of AI making decisions about people**
 - Fairness and bias
 - Transparency and explainability
 - Accountability

Example Guidelines

- Critically explore the positive and negative impacts of an AI system.
- Describe ways that AI systems can be designed for inclusivity.



Machine Bias: ProPublica.org

Big Idea #5: Societal Impact (2 of 3)

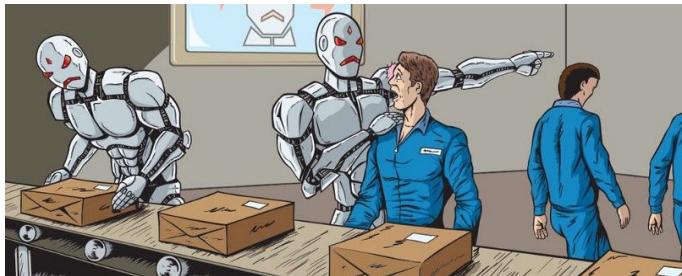
“Artificial Intelligence can impact society in both positive and negative ways.”

- **Economic impacts of AI**

- Increased productivity
- New types of services
- Reduction in of some types of jobs
- New career opportunities

Example Guidelines

- Design and explain how an AI system can be used to address a social issue.
- Understand tradeoffs in the design of AI systems and how decisions can have unintended consequences in the function of a system.



Big Idea #5: Societal Impact (3 of 3)

“Artificial Intelligence can impact society in both positive and negative ways.”

- **AI & Culture**

- Living with intelligent assistants and robot companions.
- Would you let your child travel unaccompanied in a self-driving car?
- New YouTube genre: self-driving car mishaps.

Example Guidelines

- Critically explore the positive and negative impacts of an AI system.
- Describe the debate about whether people should be polite to agents and robots.



Demos

- SpeechDemo
- Facial Landmark Detection
- FaceDemo (Deep Neural Network)
- Teachable Machine
- QuickDraw
- Machine Learning for Kids: Titanic
- TensorFlow Playground
- Calypso

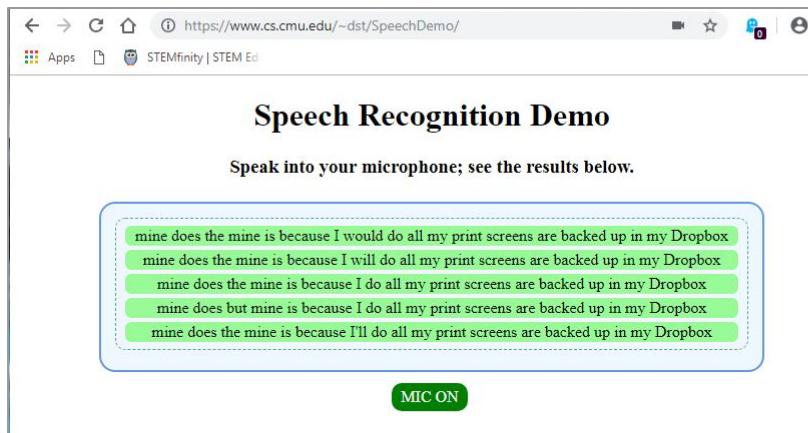
Find more great demos, videos, and curriculum resources at the AI4K12.org Resources Directory.



Speech Recognition: Speech to Text Demos



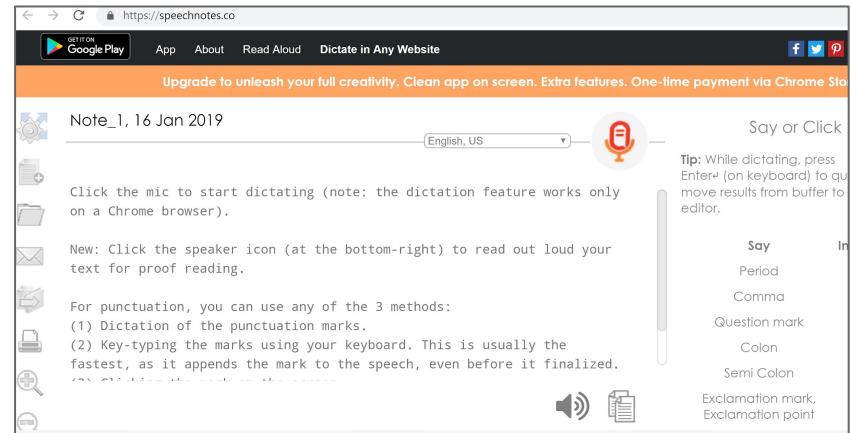
1. Demo shows alternative parses
<https://www.cs.cmu.edu/~dst/SpeechDemo/>



The screenshot shows a web browser window with the URL <https://www.cs.cmu.edu/~dst/SpeechDemo/>. The page title is "Speech Recognition Demo". Below it, a instruction says "Speak into your microphone; see the results below." A blue speech-to-text interface displays five green rectangular boxes containing the same text: "mine does the mine is because I would do all my print screens are backed up in my Dropbox". At the bottom is a green button labeled "MIC ON".



2. Demo speaks back what it heard
<https://speechnotes.co/>

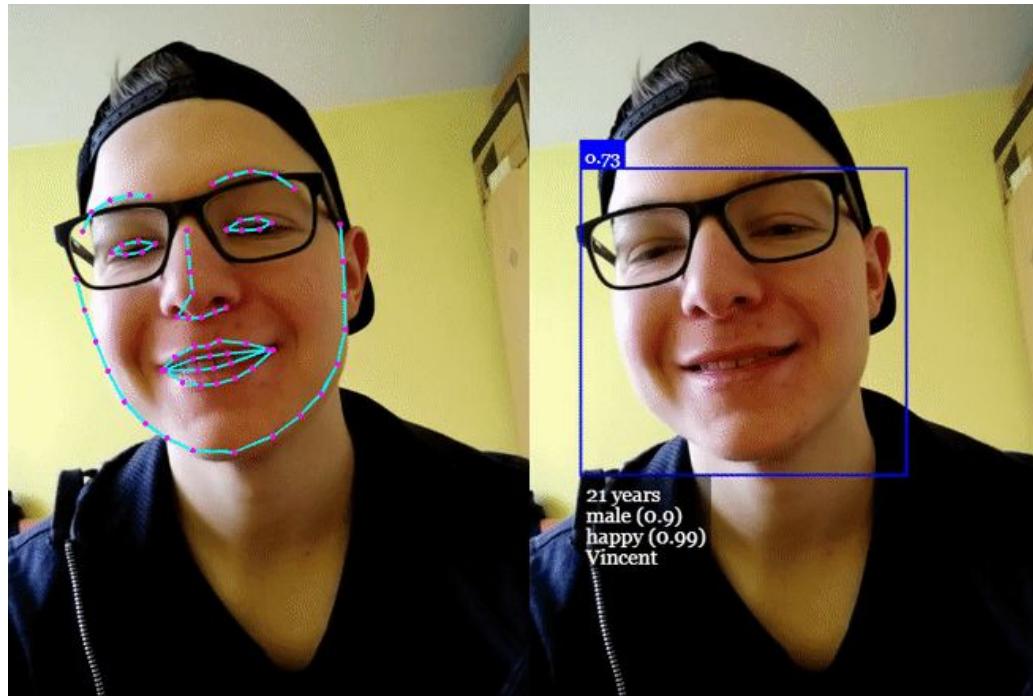


The screenshot shows a web browser window with the URL <https://speechnotes.co/>. The page title is "SpeechNotes". It features a note titled "Note_1, 16 Jan 2019" with the text "Click the mic to start dictating (note: the dictation feature works only on a Chrome browser)." Below it, instructions say "New: Click the speaker icon (at the bottom-right) to read out loud your text for proof reading." A sidebar on the right lists punctuation marks with their descriptions: Say, Period, Comma, Question mark, Colon, Semi Colon, Exclamation mark, and Exclamation point. At the bottom are icons for a speaker and a document.

Facial Landmark Recognition

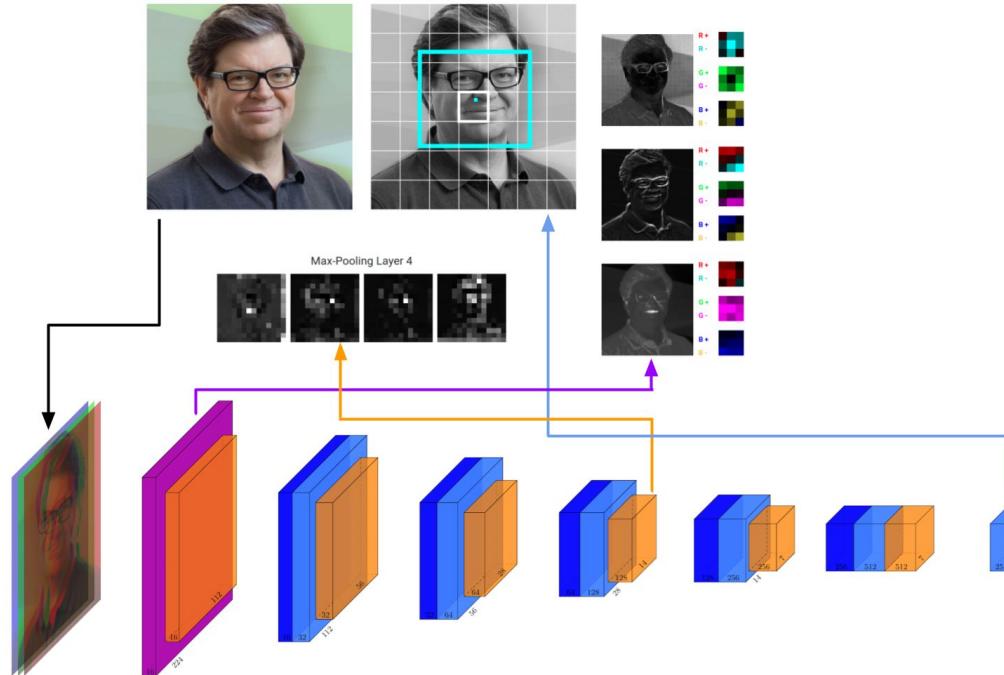
https://justadudewhohacks.github.io/face-api.js/webcam_face_tracking

Click the box for
“Detect Face
Landmarks”



Face Recognition With A Deep Neural Network

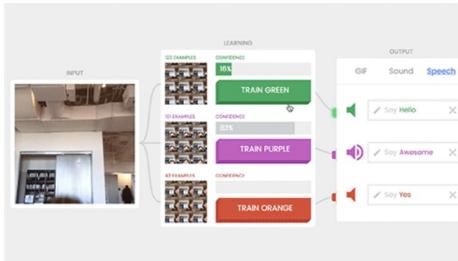
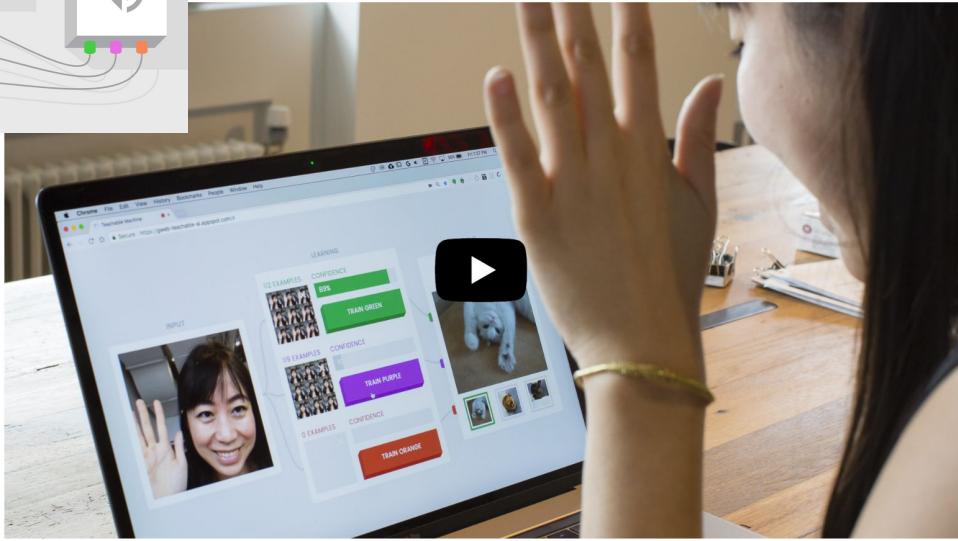
<https://www.cs.cmu.edu/~dst/FaceDemo>





Built with TensorFlow

- Teach a machine using your camera,
- live in the browser
- no coding required.

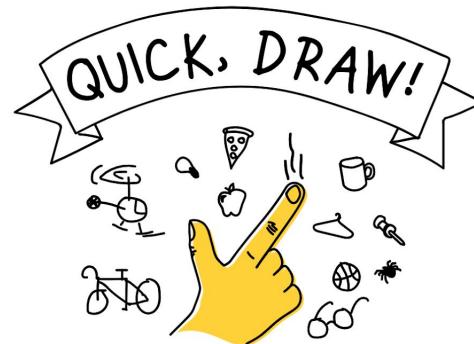


<https://experiments.withgoogle.com/teachable-machine>



Google's Quick, Draw!

<https://quickdraw.withgoogle.com/>



Can a neural network learn to recognize doodling?

Help teach it by adding your drawings to the [world's largest doodling data set](#), shared publicly to help with machine learning research.

Let's Draw!



Machine Learning for Kids

<https://machinelearningforkids.co.uk>

Machine learning projects

These projects are downloadable step-by-step guides, with explanations and colour screenshots for students to follow.

Each project is a stand-alone activity, written to last for a single lesson, and will guide children to create a game or interactive project that demonstrates a real-world use of artificial intelligence and machine learning.

Suggestions for new worksheets, suggestions of improvements to any of the worksheets, or contributions of new project worksheets, are all very welcome.

All project types

All difficulties

All make types

Smart Classroom

Create a smart assistant in Scratch that lets you control virtual devices.

Teach a computer to recognise the meaning of your commands

A screenshot of the Scratch programming environment showing a script for a 'Smart Classroom' project. The script uses various blocks to control a virtual lamp and fan. It includes conditional blocks like 'if then' and 'repeat' loops. The stage shows a simple room with a lamp and a fan.

Difficulty: Beginner
Recognising: text
Tags: digital assistants, supervised learning
[Download](#)

Make me happy

Create a character in Scratch that smiles if you say nice things to it and cries if you say mean things to it.

Teach a computer to recognise compliments and insults

A screenshot of the Scratch programming environment showing a script for a 'Make me happy' project. The script uses 'say (text)' and 'change [mood v] by (amount)' blocks. The stage features a large yellow smiley face.

Difficulty: Beginner
Recognising: text
Tags: sentiment analysis, supervised learning
[Download](#)

Snap!

Create a card game in Scratch that learns to recognise pictures of your card.

Teach a computer to recognise what icons look like

A screenshot of the Scratch programming environment showing a script for a 'Snap!' game. The script uses 'when green flag clicked' and 'when I receive [snap message v]' blocks. The stage shows two playing cards (spades) and a green background.

Difficulty: Beginner
Recognising: images
Tags: image classification, supervised learning
[Download](#)

Chameleon

Create a chameleon in Scratch that changes colour to match its background.

Teach a computer to recognise colours

A screenshot of the Scratch programming environment showing a script for a 'Chameleon' project. The script uses 'when green flag clicked' and 'when I receive [color message v]' blocks. The stage features a red chameleon on a green background.

Difficulty: Beginner
Recognising: images
Tags: image classification, supervised learning
[Download](#)

Titanic

Create a Python program that can predict who survived the sinking of the Titanic.

Teach a computer to predict outcomes

A screenshot of a Python application window titled 'Titanic'. It displays a scatter plot of survival status (Survived/Not Survived) versus age. Below the plot are several buttons labeled 'Predict', 'Clear', and 'Exit'.

Difficulty: Beginner
Recognising: numbers
Tags: predictive model, supervised learning
[Download](#)

Mailman Max

Make a postal sorting office in Scratch that can recognise handwritten postcodes on envelopes.

Teach a computer to recognise handwriting

A screenshot of the Scratch programming environment showing a script for a 'Mailman Max' project. The script uses 'when green flag clicked' and 'when I receive [sort message v]' blocks. The stage shows a green envelope with a handwritten postcode.

Difficulty: Beginner
Recognising: numbers
Tags: image classification, supervised learning
[Download](#)

Car or cup

Train the computer to be able to sort photos into groups.

Teach a computer to recognise pictures of objects

A screenshot of the Scratch programming environment showing a script for a 'Car or cup' project. The script uses 'when green flag clicked' and 'when I receive [sort message v]' blocks. The stage shows a car and a cup.

Difficulty: Beginner
Recognising: images
Tags: image classification, supervised learning
[Download](#)

Face Lock

Create a phone in Scratch that unlocks if it recognises your face.

Teach a computer to recognise faces

A screenshot of the Scratch programming environment showing a script for a 'Face Lock' project. The script uses 'when green flag clicked' and 'when I receive [unlock message v]' blocks. The stage shows a phone screen displaying a face recognition interface.

Difficulty: Beginner
Recognising: images
Tags: image classification, supervised learning
[Download](#)

Journey to school

Train the computer to be able to predict how you travel to school in the morning.

Teach a computer to make predictions

A screenshot of a Python application window titled 'Journey to school'. It displays a scatter plot of travel modes (car, walk, cycle) against time. Below the plot are buttons for 'Predict', 'Clear', and 'Exit'.

Difficulty: Beginner
Recognising: numbers
Tags: predictive model, supervised learning
[Download](#)

Shy Panda

Create a dancing panda that gets shy and stops dancing if it sees you looking.

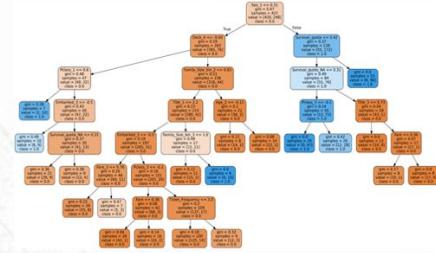
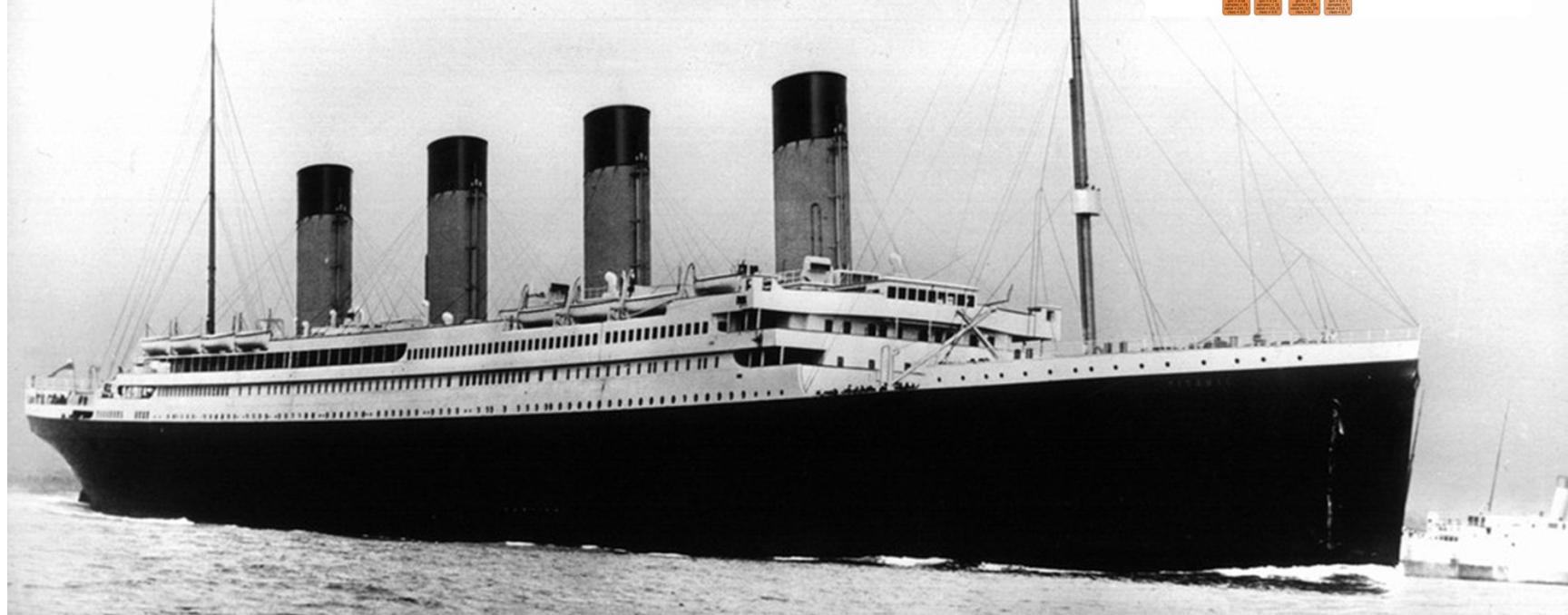
Teach a computer to recognise pictures

A screenshot of a Python application window titled 'Shy Panda'. It shows a cartoon panda character that reacts to the user's cursor. The application interface includes a 'Start' button and other controls.

Difficulty: Beginner
Recognising: images
Tags: image classification, supervised learning
[Download](#)

Machine Learning with the Titanic Dataset

An end-to-end guide to predict the Survival of Titanic passenger



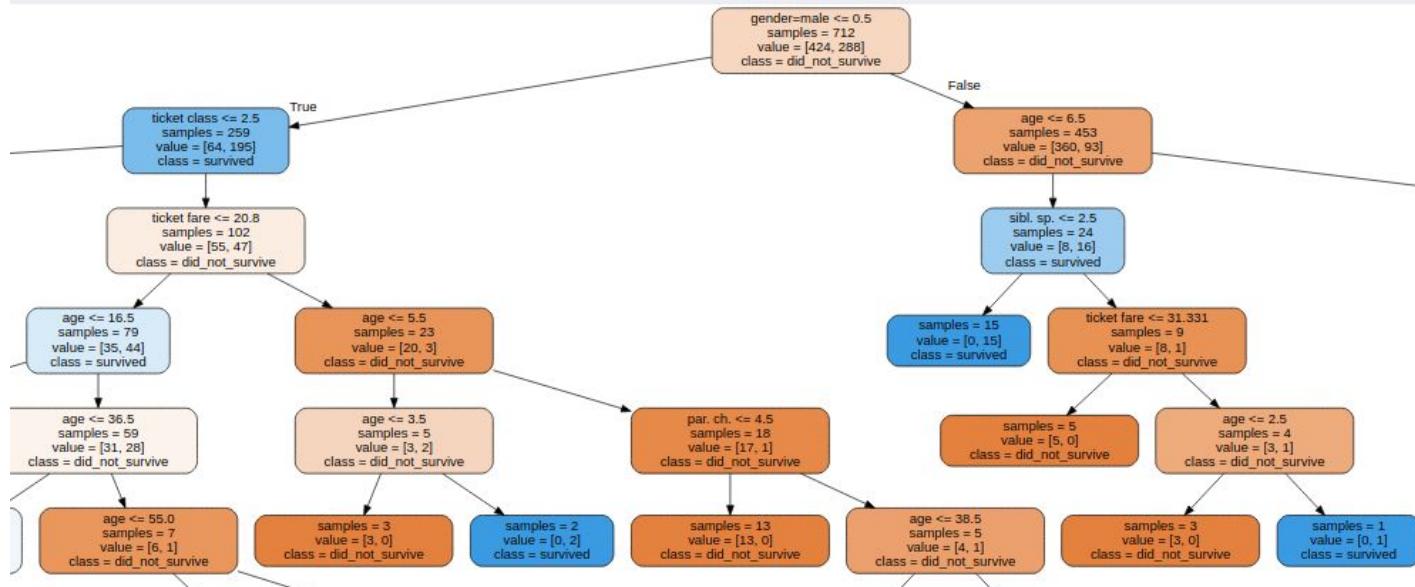
Is this test true?

samples = how many training examples got here

class = prediction so far

Go this way if
the test is true

Go this way if
the test is false



TensorFlow Playground

<https://playground.tensorflow.org>



Epoch 000,000 Learning rate 0.03 Activation Tanh Regularization None Regularization rate 0 Problem type Classification

DATA
Which dataset do you want to use?
 MNIST
 Fashion MNIST
 Flowers
Ratio of training to test data: 50%
Noise: 0
Batch size: 10

FEATURES
Which properties do you want to feed in?
 x_1 x_2 x_1^2 x_2^2 $x_1 x_2$ $\sin(x_1)$ $\sin(x_2)$

+ - 2 HIDDEN LAYERS
4 neurons 2 neurons
This is the output from one neuron. Hover to see it larger.
The outputs are mixed with varying weights, shown by the thickness of the lines.

OUTPUT
Test loss 0.508
Training loss 0.504

Colors shows data, neuron and weight values.
 Show test data Discretize output

REGENERATE

Tutorial: <https://cloud.google.com/blog/products/gcp/understanding-neural-networks-with-tensorflow-playground>



Calypso for Cozmo



- A robot intelligence framework that combines multiple types of AI:
 - Computer vision
 - Speech recognition
 - Landmark-based navigation
 - Path planning
 - Object manipulation
- Rule-based language inspired by Microsoft's Kodu Game Lab
- Teaches computational thinking
- Web sites:
 - <https://Calypso.software> (Cozmo robot version)
 - <https://calypso-robotics.com> (free simulator version runs in the browser)

Calypso 0.9.04 - Google Chrome

15-294-A3 Rapid Prototype | 15-494/694 (5 unread) | WAICY 2019 Rubric - Read | Calypso 0.9.04 | +

127.0.0.1:43125/Calypso/index.html

Apps West Mifflin, PA Browse K-12 STE Seniors For Safe CMU Oracle Web S3 Admin Console Home - Workday The Best of the F 15-294-A3 Rapid Cognitive Robot

Stop program State machine view Switch characters Map editor view Stop program Scroll up/down

1

WHEN see cube 1 DO move toward it

WHEN bumped cube 1 DO grab it

WHEN + DO switch to page 2

rules

speech recognition

perception

world map

LightCube 3 id=1 LightCube 2 id=2 id=44 id=45

Calypso 0.9.04

Cozmo's battery 3.9 volts Cube3 batt 1.24V (48%)

0

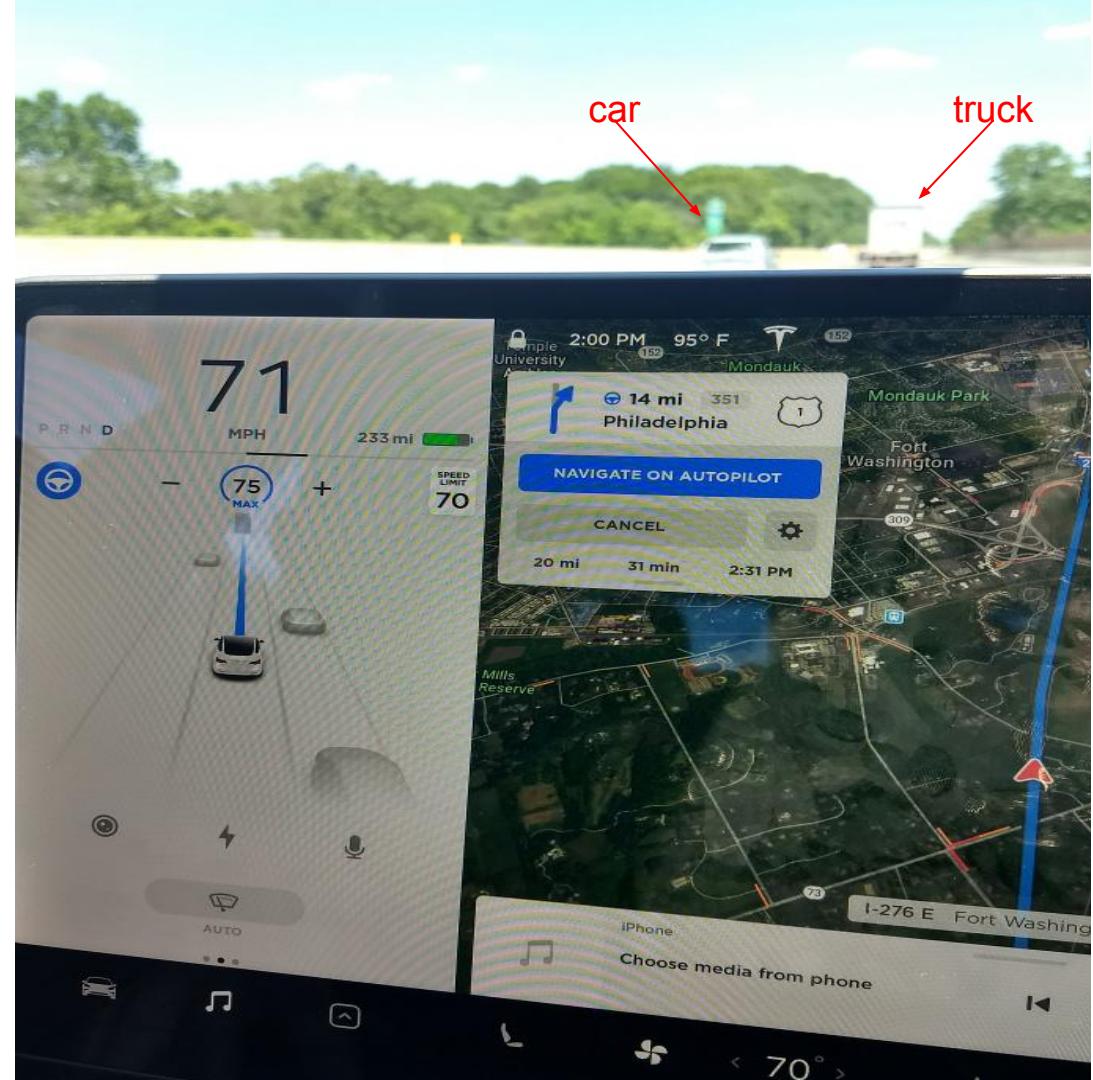
34

The screenshot displays the Calypso 0.9.04 software interface. On the left, a script editor window titled '1' shows three Scratch-like scripts. The first script uses 'WHEN see cube 1' to make the robot move towards and grab a cube. The second script uses 'WHEN bumped cube 1' to make the robot grab the cube. The third script is a conditional loop starting with 'WHEN +' that switches to 'page 2'. A pink callout box labeled 'rules' points to the script editor. On the right, a large orange speech bubble says 'hello cozmo', with a pink callout box labeled 'speech recognition' pointing to it. Below the bubble, two small cubes are shown. At the bottom left, a camera feed shows three light cubes labeled 'LightCube 3 id=1', 'LightCube 2 id=2', and 'LightCube 1 id=44'. Another light cube labeled 'LightCube 1 id=45' is also visible. A pink callout box labeled 'perception' points to this camera feed. At the bottom right, a world map is shown with a pink callout box labeled 'world map'. The status bar at the bottom indicates 'Cozmo's battery 3.9 volts' and 'Cube3 batt 1.24V (48%)'. The top navigation bar shows tabs for various applications like 'West Mifflin, PA', 'Browse K-12 STE', and 'Seniors For Safe'.

Tesla's World Map

Cozmo can be considered “a self-driving car in the palm of your hand.”

At right is an image from a real self-driving car, a Tesla, showing the road and other nearby vehicles on its world map.



**Join Us in Developing the Guidelines, or Help
Grow the Community of AI Resource Developers**

Visit us:

<http://AI4K12.org>

Join the mailing list:

<https://aaai.org/Organization/mailing-lists.php>



Questions?

Thank
You!

