PRINCETON AUTONOMOUS VEHICLE ENGINEERING

Deep Learning for Self-driving Car

Chenyi Chen

Background

- Design and test a new algorithm on a real car is:
- Time consuming to set up everything
- Not very safe
- Less convenient
- Most important: Prospect 12 is down!
- So, let’s try to use a racing game!

Background

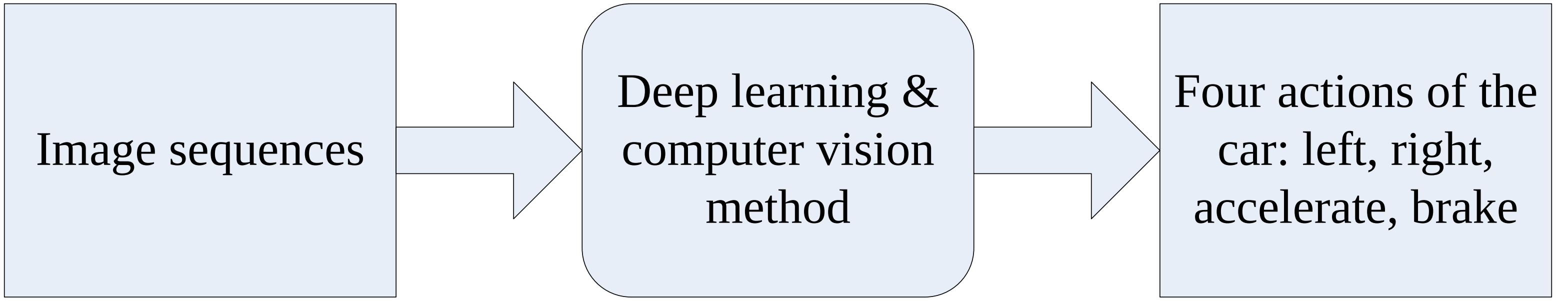
- Let the deep learning vision algorithm drive in a racing game -- TORCS

Why TORCS? Not Need for Speed?

- Open source, so you can access the source code (most important)
- Widely used in artificial intelligence research community
- Good vehicle dynamics and mechanics model
- Easy to start with
- Run on Linux

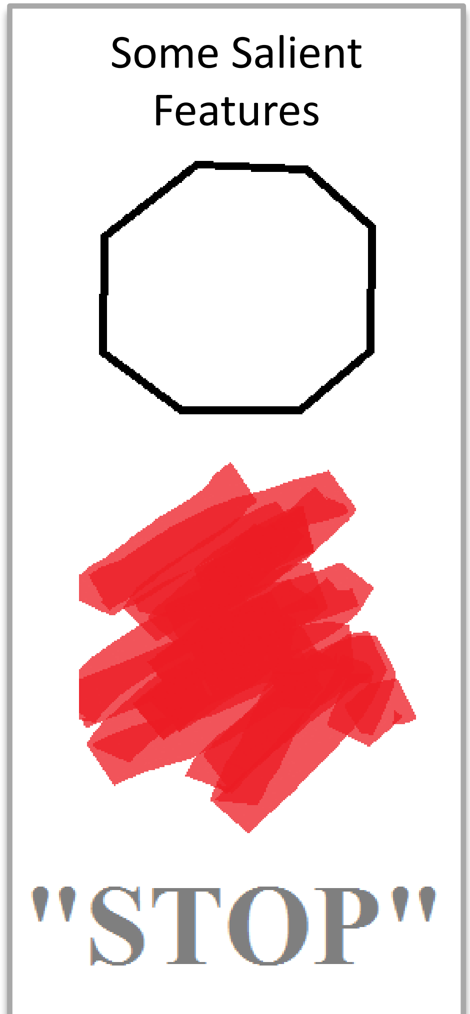
Original Version: Basic Idea

- Mapping images to driving actions



Why deep learning?

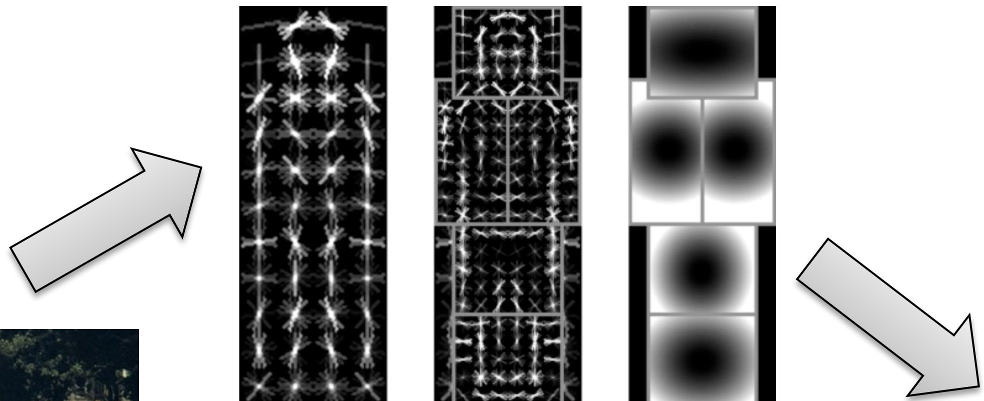
How do we detect a stop sign? It's all about feature!



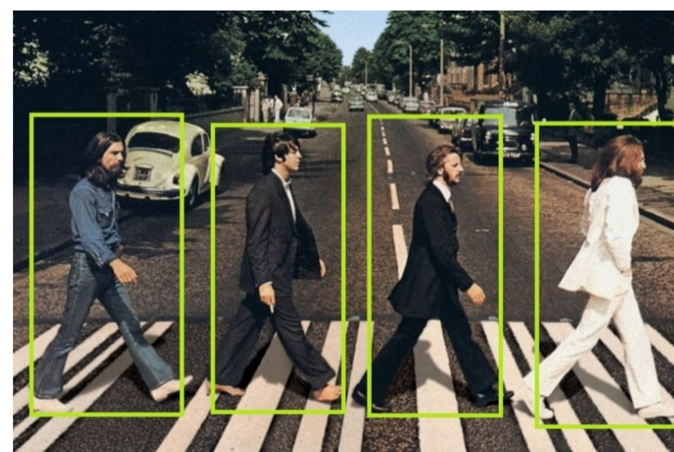
It's a STOP sign!!

Why deep learning?

How does computer vision algorithm work? It’s all about feature!

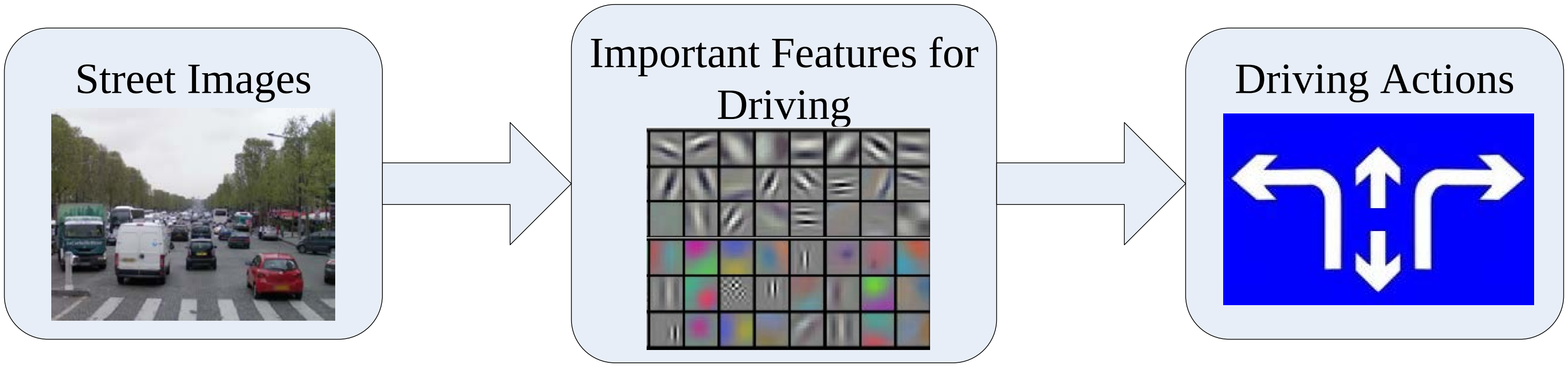


Pedestrian found!!!



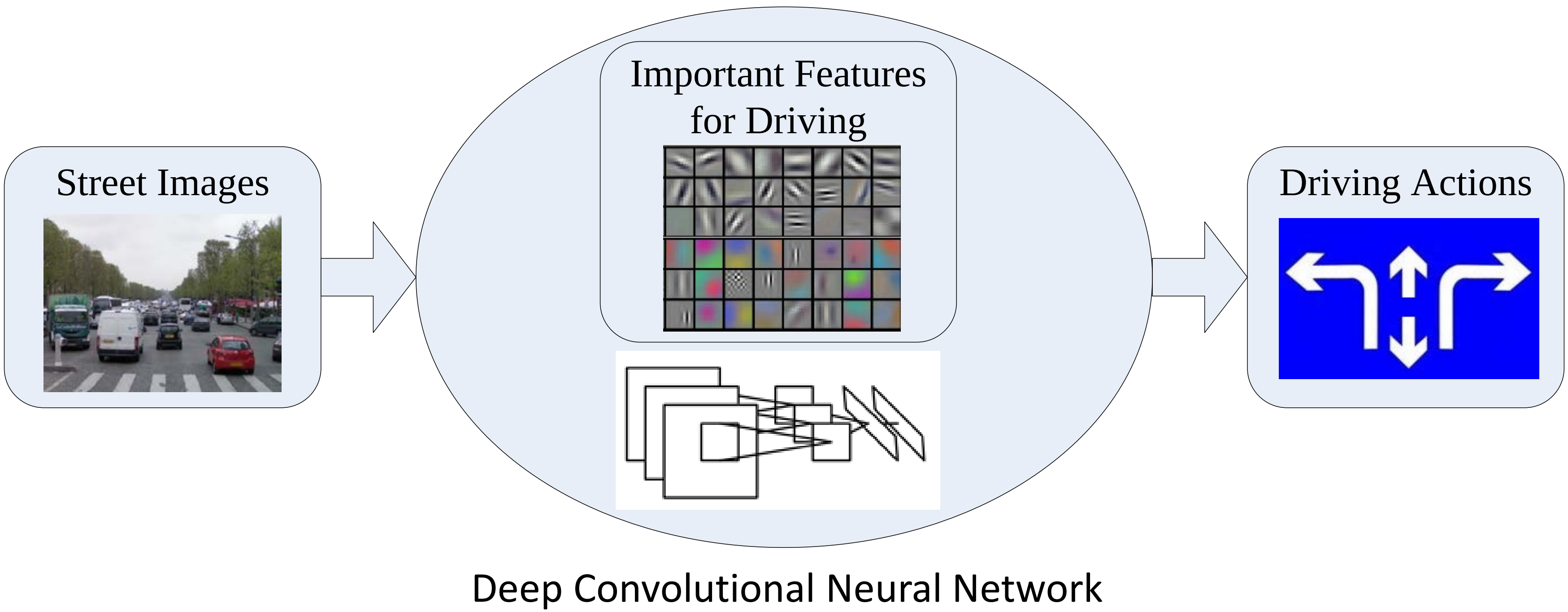
Why deep learning?

- We believe driving is also related with certain features
- Those features determine what action to take next



Why deep learning?

- Salient features can be automatically detected and processed by deep learning algorithm
- A mapping between features and actions is established during training



Why deep learning?

- ImageNet Classification Challenge
  - 1000 categories
  - 1.2 million training images
  - 50,000 validation images
  - 150,000 testing images
  - Top-5 error rate* of deep learning: 15.3%
  - Top-5 error rate of second best (which is non-deep learning): 26.2%

***Top-5 error rate**: the fraction of test images for which the correct label is not among the five labels considered most probable by the model

Deep Convolutional Neural Network (CNN)

- The network structure that achieved the excellent performance in ImageNet Classification Challenge

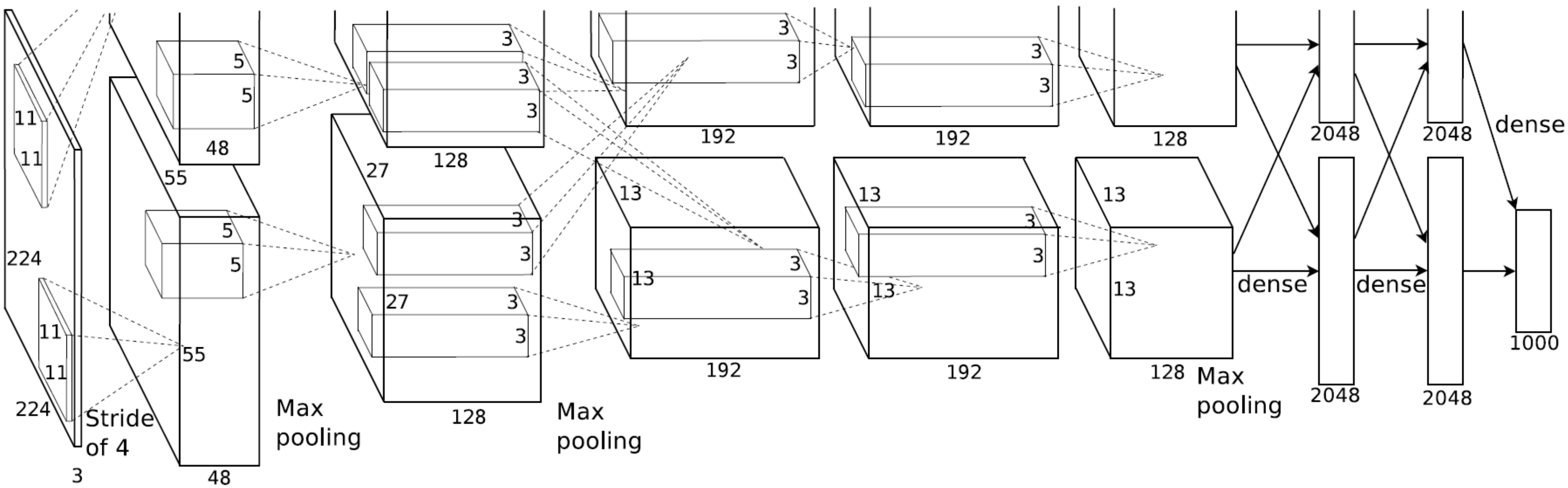
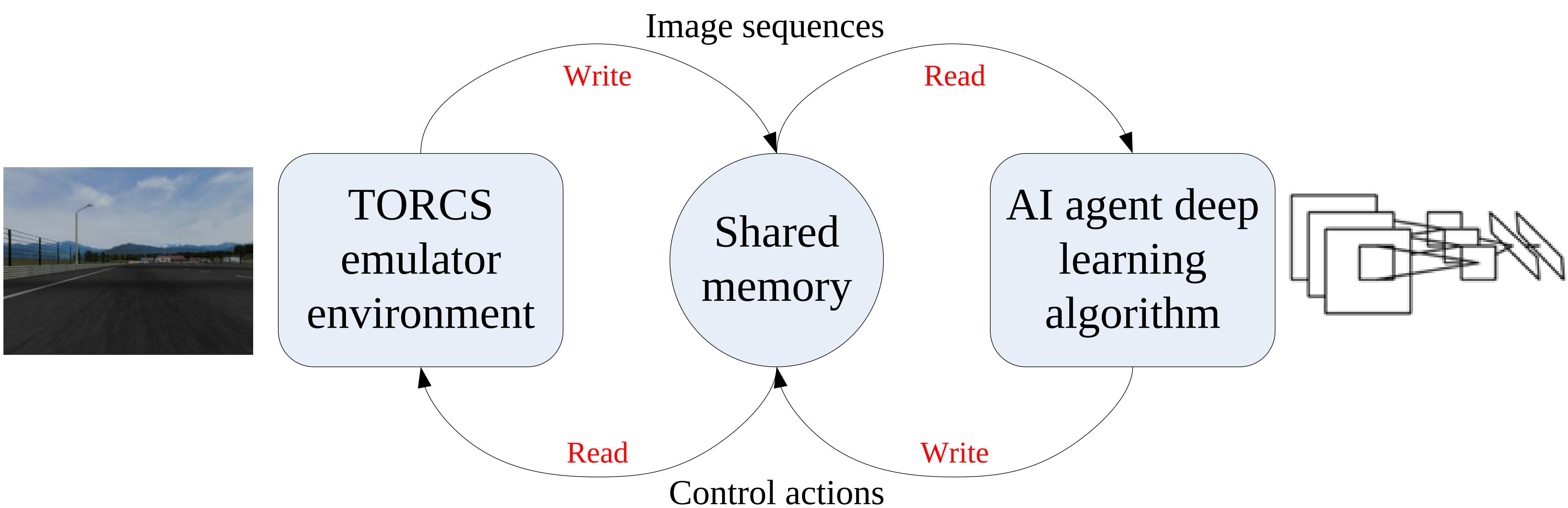


Figure courtesy of Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton

Communications between the TORCS and the AI Agent

- Memory Sharing

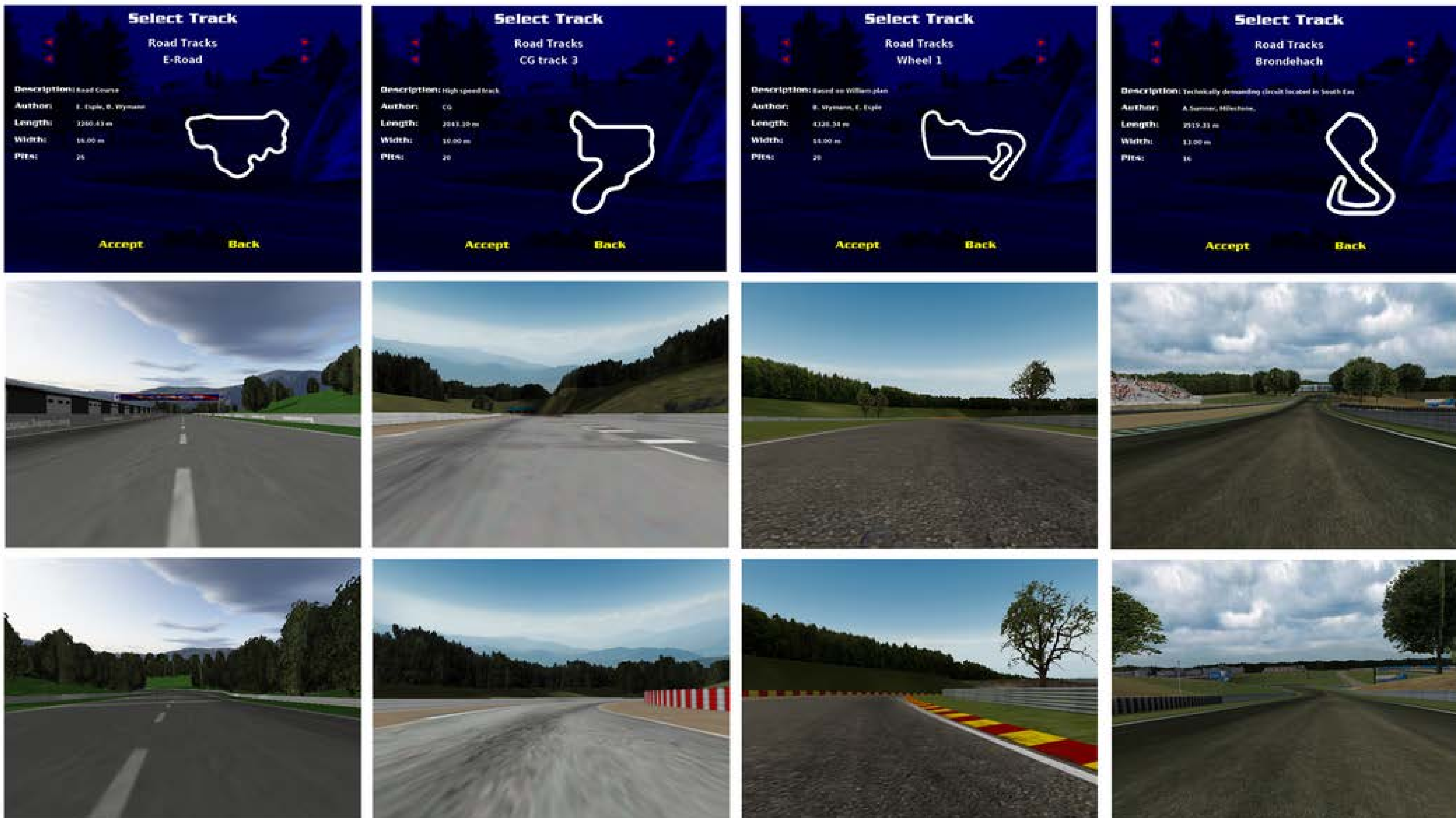


1st step: driving without other vehicles

- Simplified scene
- No complicated motion (cause no other cars)
- Much easier task
- All the information needed for driving can be encoded in a single frame

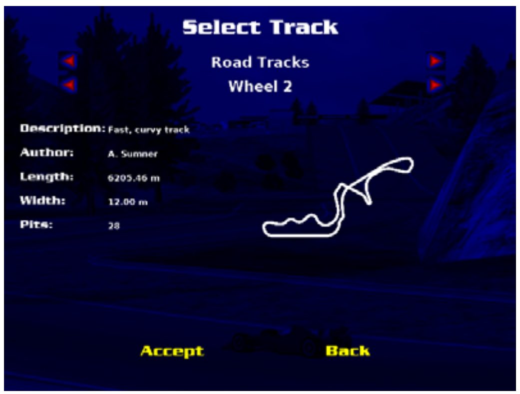
Four Tracks Used as Training Set

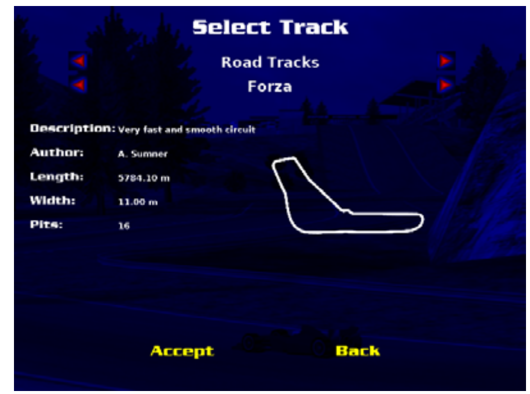
- The images (x) and the corresponding driving controls (y) are recorded



Six Tracks Used as Testing Set

- No overlap with the training set





What does the algorithm do?

- Process the image
- Output steering command based on image content
- Also output desired speed for current road condition
- Feedback the actual speed of the car, and let a speed controller to control the throttle/brake

Test track: Wheel 2

What’s next?

- Of course, drive in traffic
- Goal: stay on the track & avoid collision
- Problem: driving with and without other cars are two totally different problems
- Complicated motion is involved
- Challenging machine learning task

But this time, direct learning sucks!

Why?

- Millions of driving scenes, only four types of controls: left, right, accelerate, brake
- We human can do reasoning to differentiate diverse driving scenes and map them to the four actions, but machine learning cannot
- So it’s too difficult for machine learning algorithms to learn driving controls directly from complicated driving scenes

So, let’s make the task easier for our poor algorithm

How?

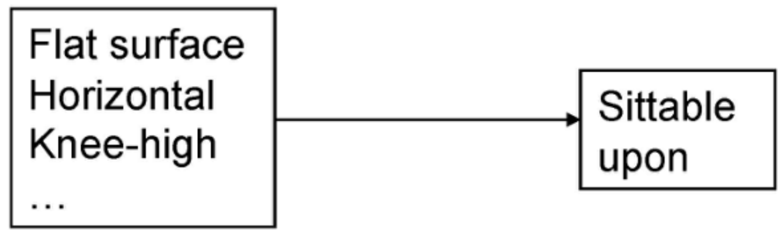
- Extract key parameters from driving scenes with deep learning
- Compute driving control (optimal control) based on those parameters

Direct Perception

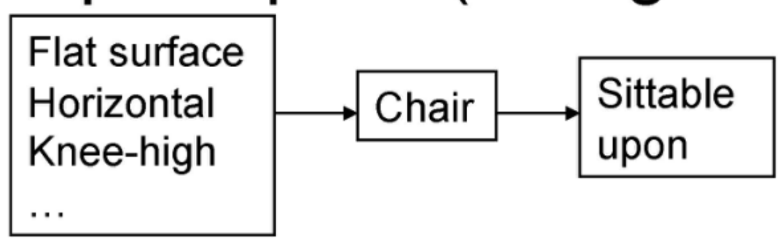
- We can perceive the 3D shape, texture, material properties, without knowing the category of objects.
- But the category of objects also encapsulates about what can we do with the objects

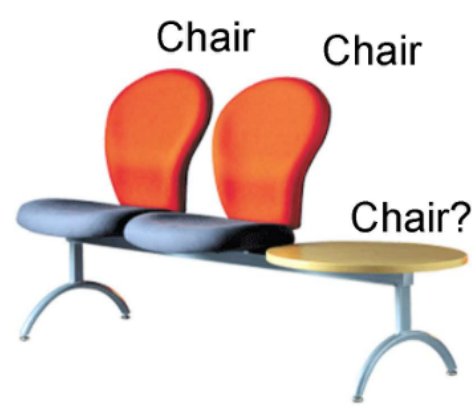
The perception of function

- Direct perception (affordances):



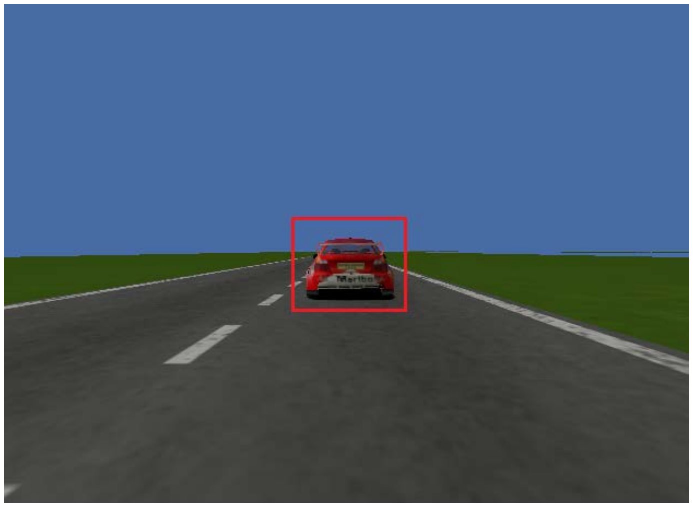
- Mediated perception (Categorization)





Coutesy of Antonio Torralba

In Driving…



- Ordinary car detection: Find a car! It's localized in the image by a red bounding box.
- Direct perception: The car is in the right lane; 16 meters ahead
- Which one is more helpful for our driving task?

In Our Specific Case…

- Let the deep learning algorithm tell us:
- angle: the angle between the car's heading and the tangent of the track;
- toMiddle: the distance between the center of the car and the center line of the track;
- min_dist: the distance between the car and the 1st preceding car;
- lane: the lane of the 1st preceding car

Where to get the training data?

- Training data of the four parameters are collected from the game engine
- For real images, we can get the labels through crowdsourcing, e.g. Amazon Mechanical Turk
- Or measure such parameters with special equipment when collecting the data, e.g. Google Street View
- Or more crazily, train on simulation, test on real car!

Then there comes the demo

So, what's next?

Wow, something real!

Wow, something real!

Q & A