Distance Estimation

Overview

- **Perspective Projection**
- **The Multi-Layer Perception:** Learning with Backpropagation
- **Example (Code):**
  Accurate Distance Estimation to (Possibly Moving) Objects

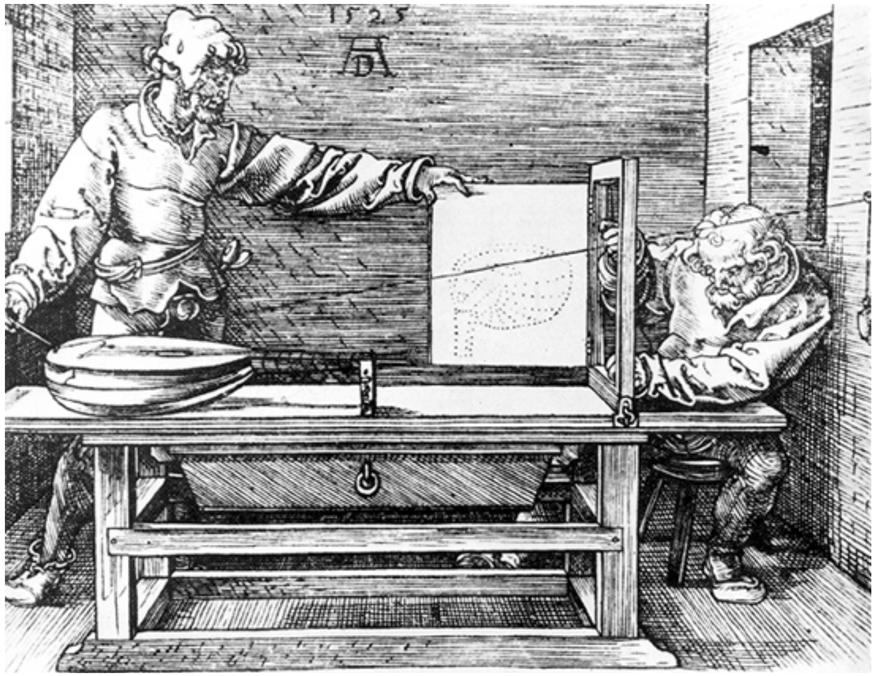
Robot Vision: Discovering Perspective Projection

Egyptian painting (15th century BC)
Source: Wikimedia Commons

Ancient Egyptian wall paintings:

- "Flat" - two dimensional art
  - Depth is shown by overlapping figures
  - Size of objects indicate their importance

Robot Vision: Discovering Perspective Projection



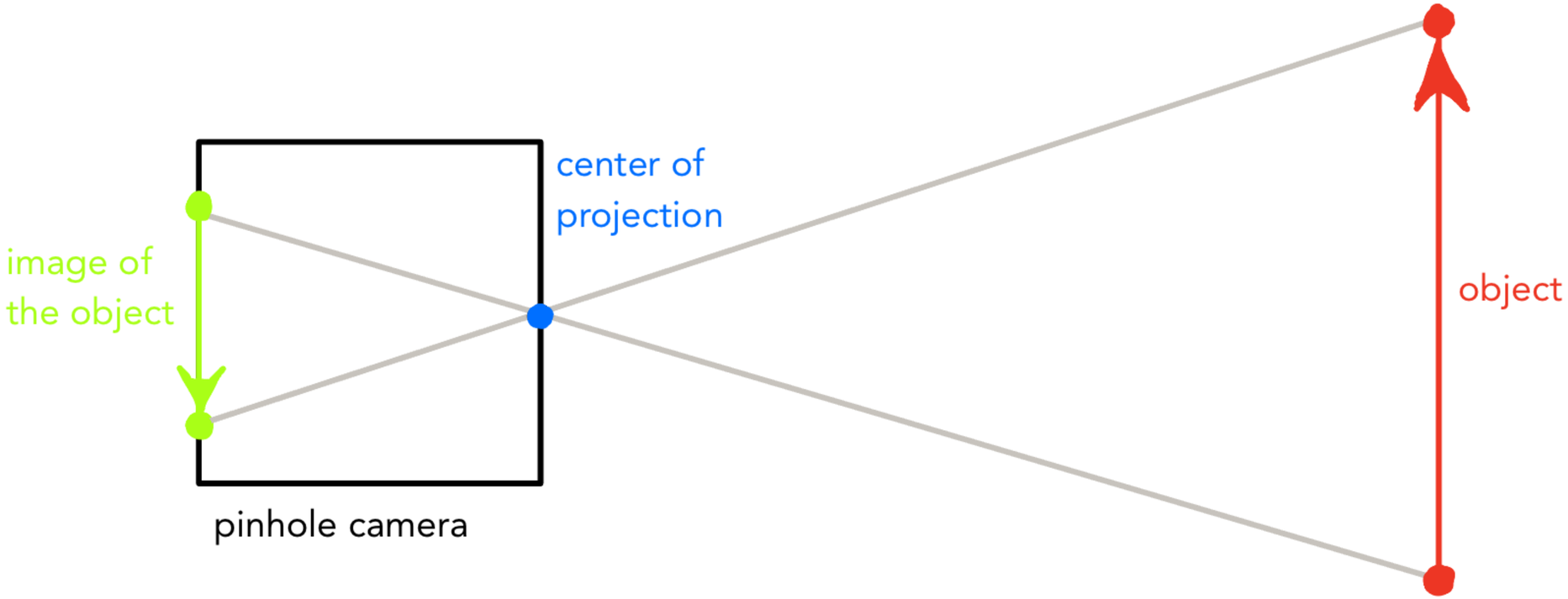
Man drawing a lute, Albrecht Dürer, 1525
Source: Wikimedia Commons

Renaissance Art:

- Linear perspective projection
  - "Constructed" geometrical mapping of the world on a two-dimensional surface creates the illusion of depth
  - Size of objects indicate their distance

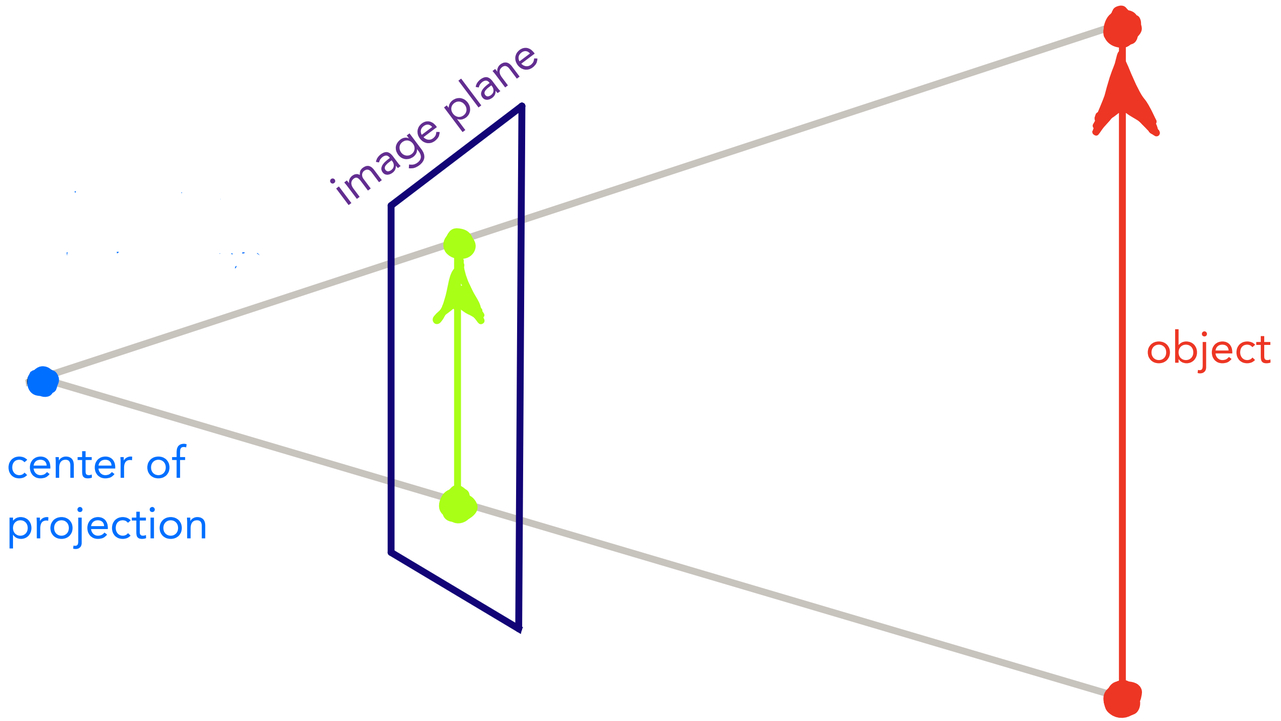
Robot Vision: Discovering Perspective Projection

Perspective Projection:
Linear mapping of the 3D world onto the camera sensor (or image plane).



Robot Vision: Discovering Perspective Projection

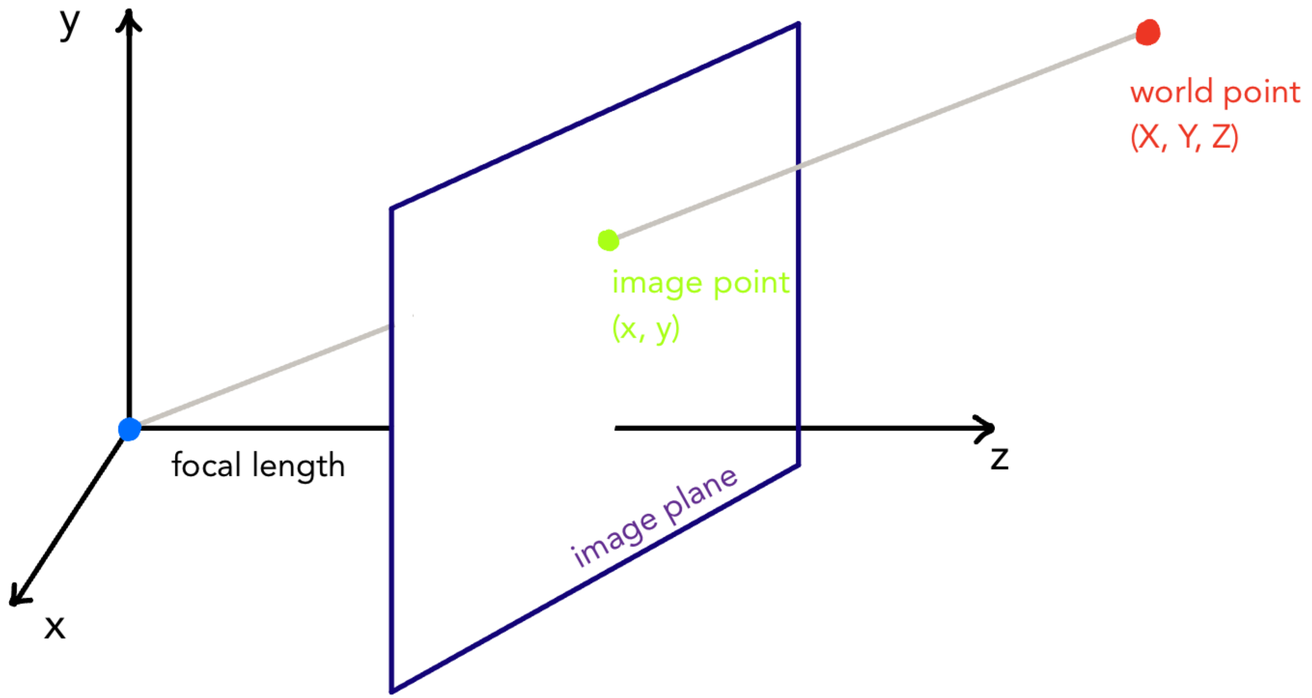
Perspective Projection:
Linear mapping of the 3D world onto the camera sensor (or image plane).



Robot Vision: Discovering Perspective Projection

Perspective Projection:
Linear mapping of the 3D world onto the camera sensor (or image plane).

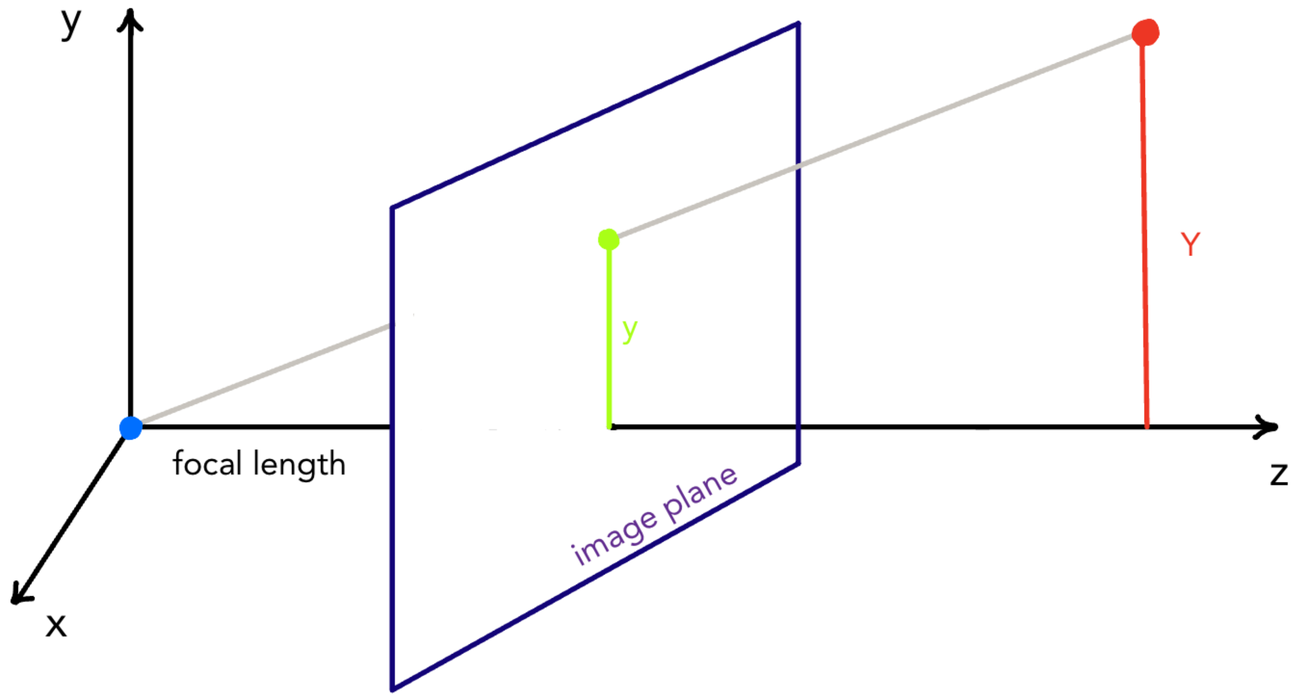
$$y = f\frac{Y}{Z} + c_y \qquad x = f\frac{X}{Z} + c_x$$



Robot Vision: Discovering Perspective Projection

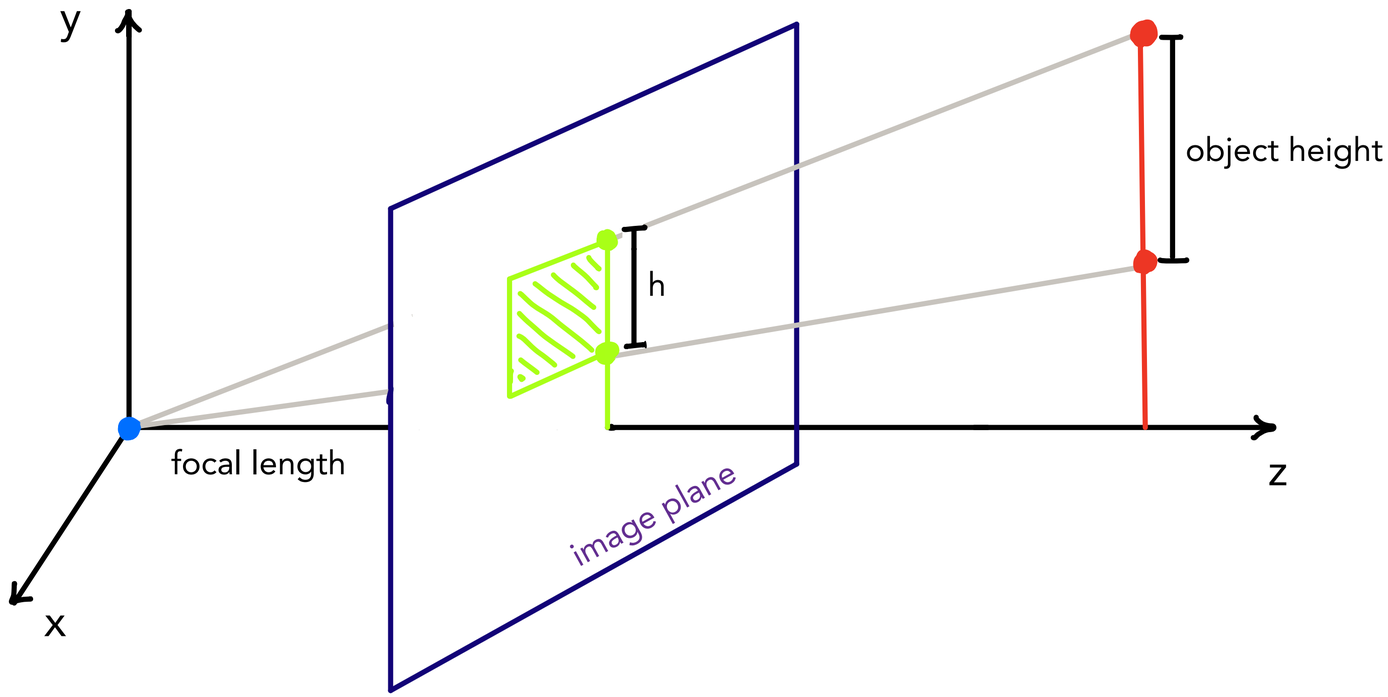
Perspective Projection:
Linear mapping of the 3D world onto the camera sensor (or image plane).

$$y = f\frac{Y}{Z} + c_y \qquad x = f\frac{X}{Z} + c_x$$



Question:
How to infer the distance to an object?

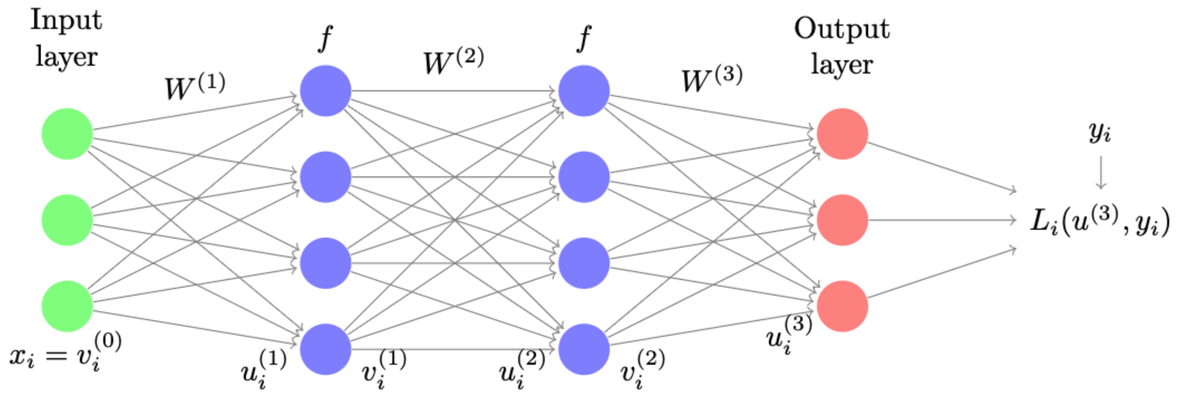
Given the bounding box of an object on the image plane, the height of the real object and the camera intrinsics, how can we compute the distance to an object?



Overview

- **Perspective Projection**
- **The Multi-Layer Perception:** Learning with Backpropagation
- **Example (Code):**
  Accurate Distance Estimation to (Possibly Moving) Objects

The Multilayer Perceptron



The multilayer perceptron (MLP) is an artificial neural network, consisting of fully connected nodes with a nonlinear activation function. The MLP is capable of learning a non linear function between input and output data.

Learning with Backpropagation

The MLP in equations:

$$u^{(1)} = W^{(1)} \cdot v^{(0)}$$
$$v^{(1)} = f(u^{(1)})$$
$$u^{(2)} = W^{(2)} \cdot v^{(1)}$$
$$v^{(2)} = f(u^{(2)})$$
$$u^{(3)} = W^{(3)} \cdot v^{(2)}$$
$$L = \frac{1}{N} \sum_i L_i(u_i^{(3)}, y_i)$$

How to change the weights of each layer, to minimise the loss L?

Learning with Backpropagation

In order to update the weights W we compute the derivatives $\frac{\partial L}{\partial W^{(3)}}$, $\frac{\partial L}{\partial W^{(2)}}$ and $\frac{\partial L}{\partial W^{(1)}}$ using back propagation

Learning with Backpropagation

$$\begin{aligned}\frac{\partial L}{\partial W^{(3)}} &= \frac{\partial L}{\partial u^{(3)}} \cdot \frac{\partial u^{(3)}}{\partial W^{(3)}} \\ &= \frac{\partial L}{\partial u^{(3)}} \cdot v^{(2)} \\ &= \delta^{(3)} \cdot v^{(2)}\end{aligned}$$

$$\begin{aligned}\frac{\partial L}{\partial W^{(2)}} &= \frac{\partial L}{\partial u^{(3)}} \cdot \frac{\partial u^{(3)}}{\partial v^{(2)}} \cdot \frac{\partial v^{(2)}}{\partial u^{(2)}} \cdot \frac{\partial u^{(2)}}{\partial W^{(2)}} \\ &= \delta^{(3)} \cdot W^{(3)} \cdot \frac{\partial v^{(2)}}{\partial u^{(2)}} \cdot v^{(1)} \\ &= \delta^{(2)} \cdot v^{(1)}\end{aligned}$$

$$\begin{aligned}\frac{\partial L}{\partial W^{(1)}} &= \frac{\partial L}{\partial u^{(3)}} \cdot \frac{\partial u^{(3)}}{\partial v^{(2)}} \cdot \frac{\partial v^{(2)}}{\partial u^{(2)}} \cdot \frac{\partial u^{(2)}}{\partial v^{(1)}} \cdot \frac{\partial v^{(1)}}{\partial u^{(1)}} \cdot \frac{\partial u^{(1)}}{\partial W^{(1)}} \\ &= \delta^{(2)} \cdot W^{(2)} \cdot \frac{\partial v^{(1)}}{\partial u^{(1)}} \cdot v^{(0)} \\ &= \delta^{(1)} \cdot v^{(0)}\end{aligned}$$

Question:

How to learn the distance to an object from data?

DO IT YOURSELF