Welcome to CMPSCI 574 / 674!

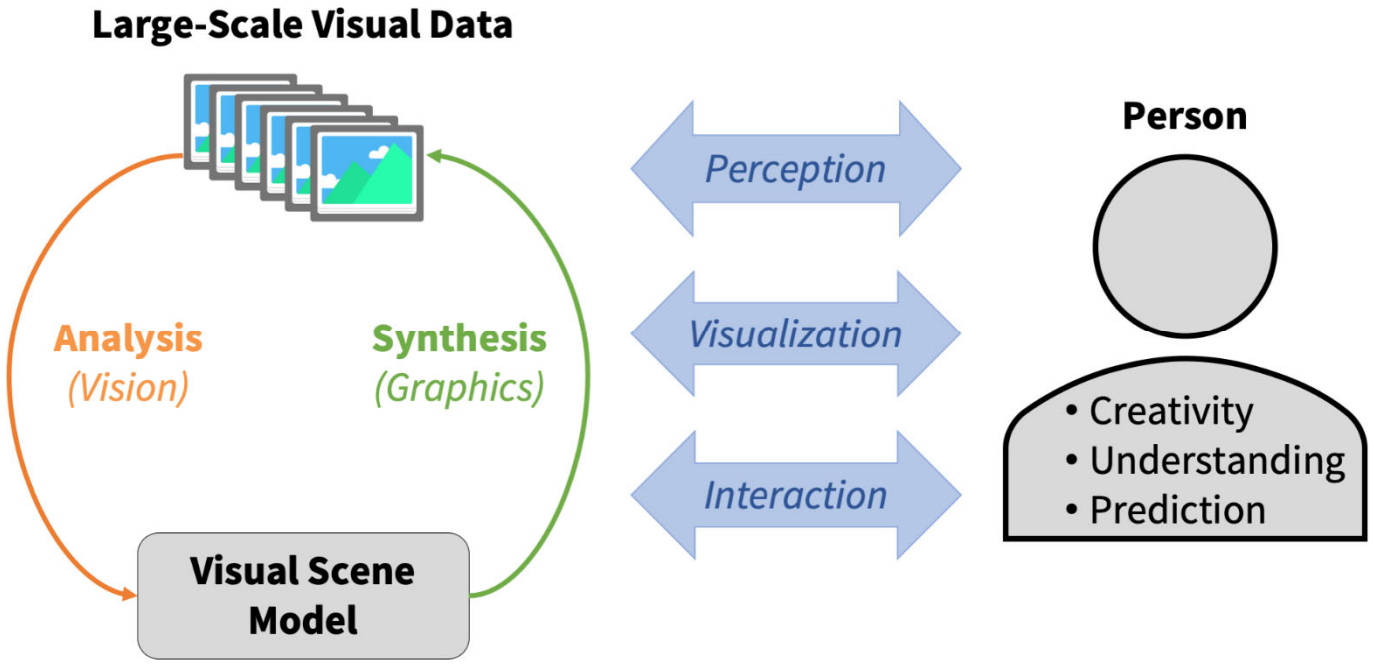
Intelligent Visual Computing: A neural network approach

Instructor:
Evangelos Kalogerakis

Visual Computing

Visual computing is about *"**making**"* and "***making sense of***" **visual data:**

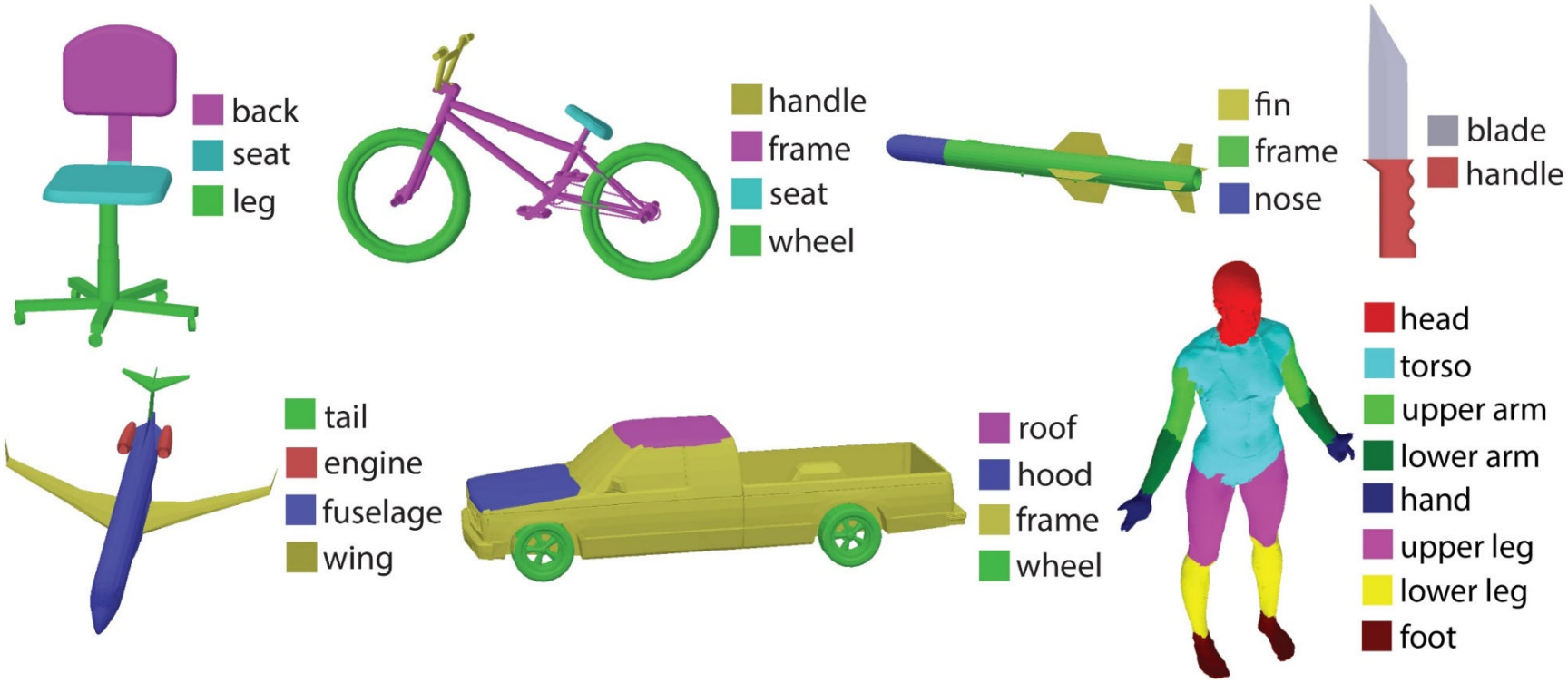
- ***analysis*** methods (i.e. computer vision) to extract rich models from visual data (e.g. images, 3D shapes, point clouds),
- **synthesis** methods (i.e. computer graphics) to convert those models back into observable visual data.



https://visual.cs.brown.edu/

“Intelligent” Visual Computing

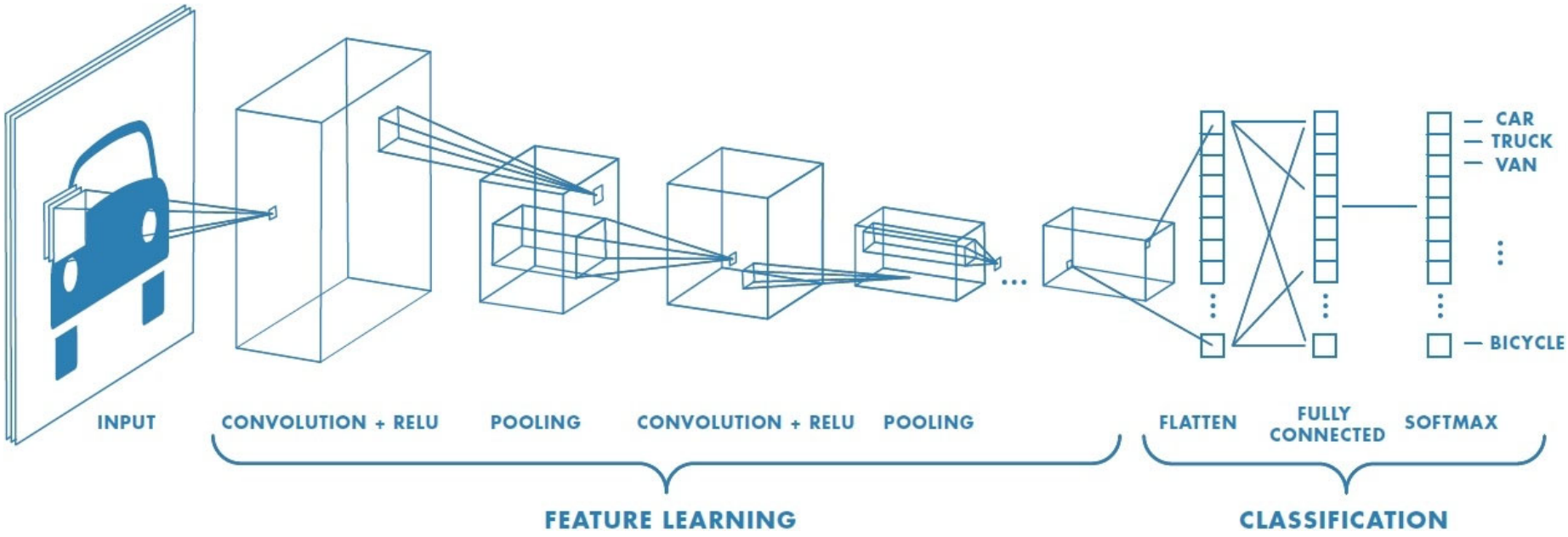
Several complex tasks require **intelligent** processing to output results
... similar to what humans would expect



3D Shape Segmentation with Projective Convolutional Networks, Kalogerakis et al, CVPR 2017

Key Idea for “Intelligent” Visual Computing

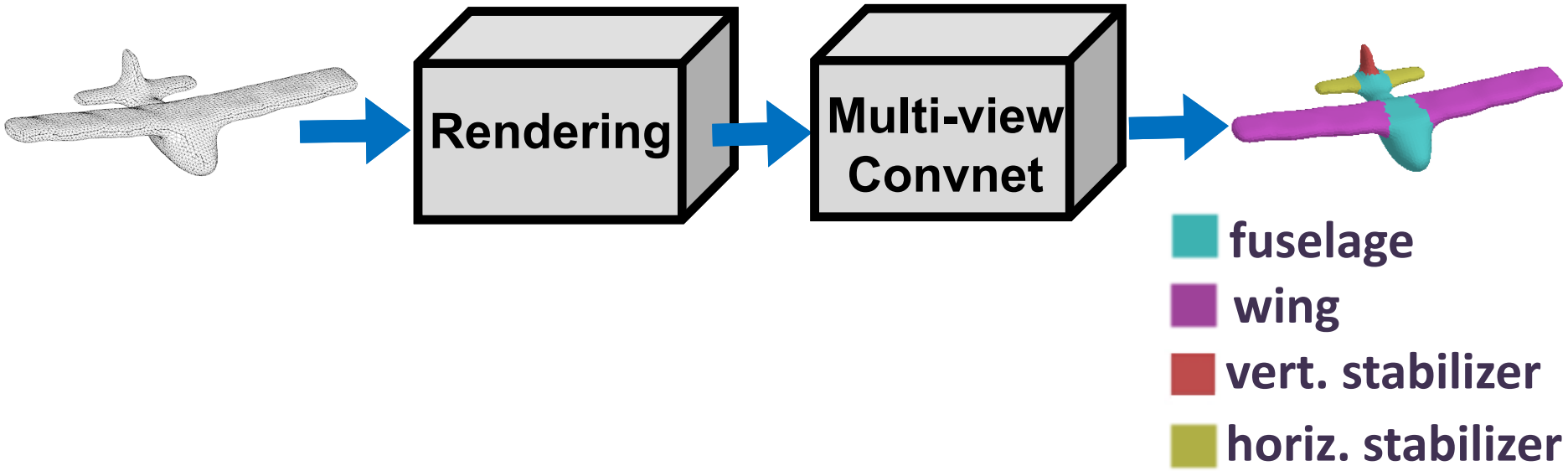
Use **neural networks:** convolutional neural networks, transformers, generative adversarial networks, diffusion models etc



Source: Mathworks

Differences from CMPSCI 682/670

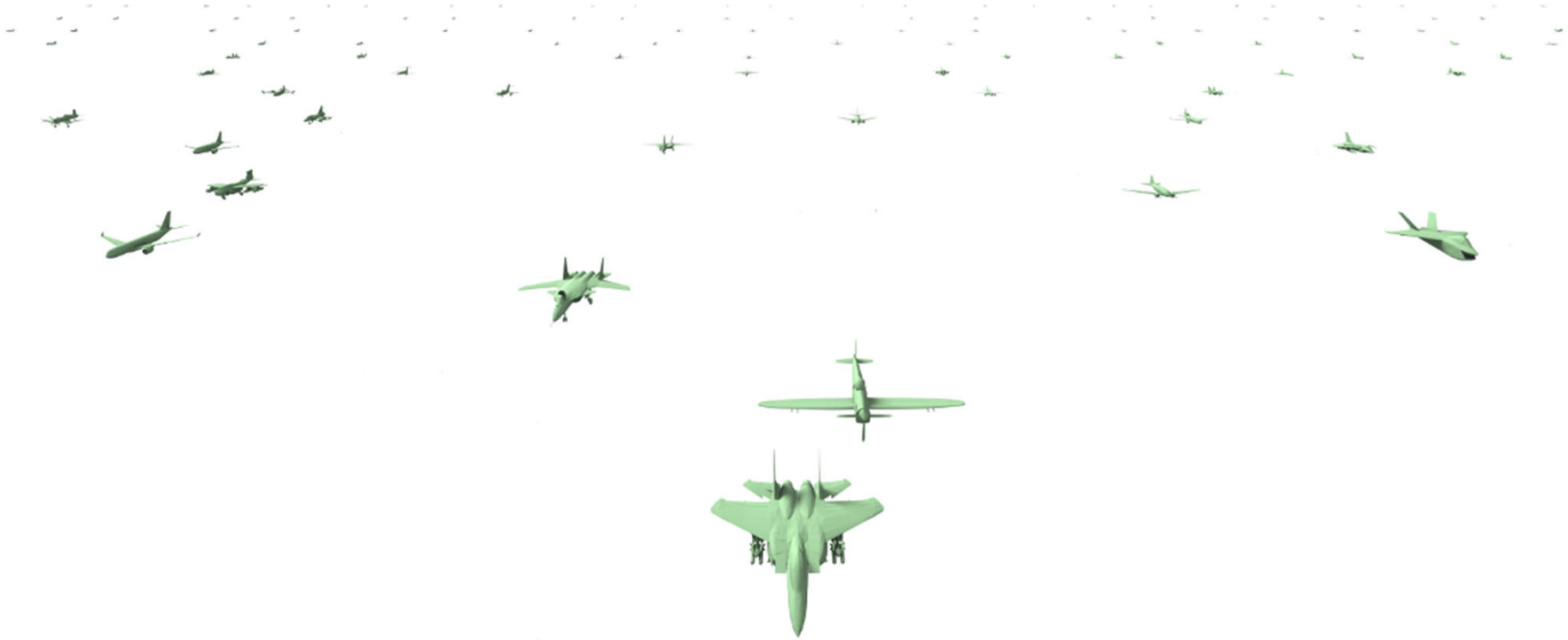
Combine neural networks with **models and principles** used in computer graphics (rendering, animation, geometry) for best results



3D Shape Segmentation with Projective Convolutional Networks, Kalogerakis et al, CVPR 2017

Differences from CMPSCI 682/670

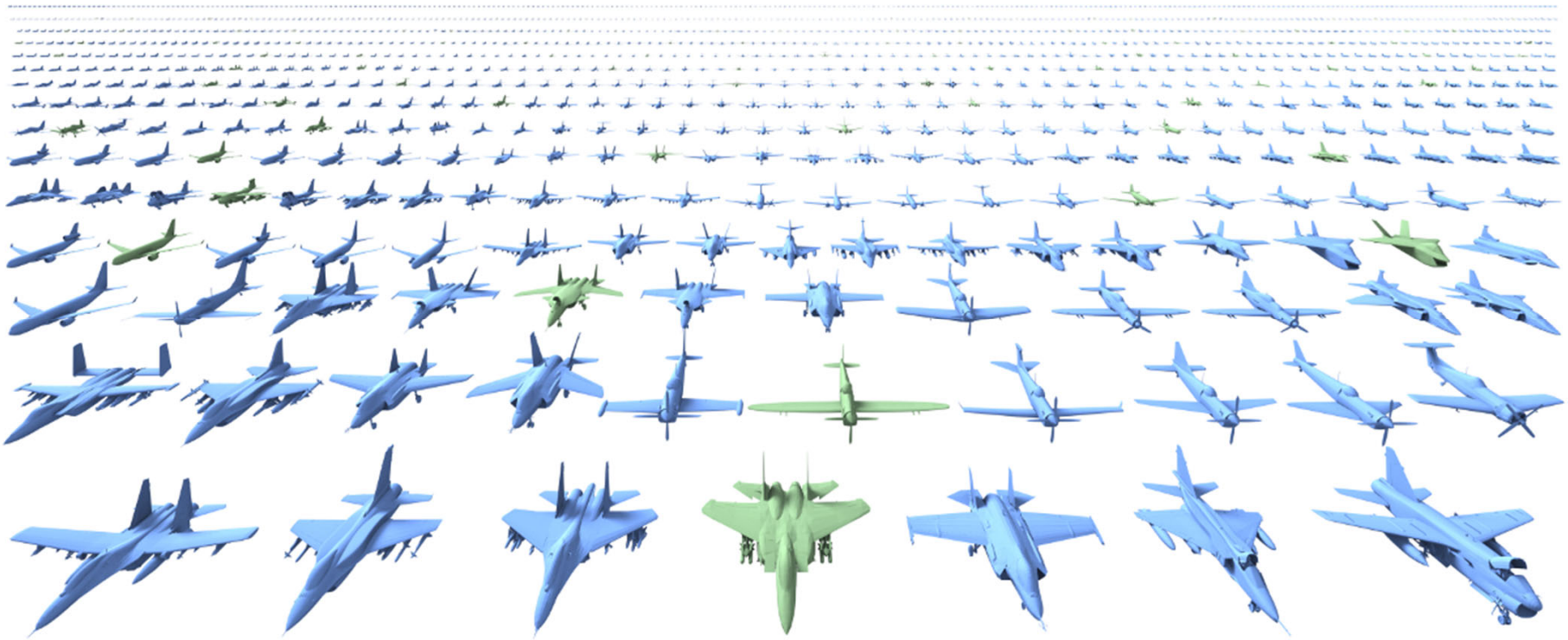
A large part of the course focuses on **3D visual data analysis & synthesis**



Kalogerakis et al, A probabilistic model for component-based shape synthesis, SIGGRAPH 2012

Differences from CMPSCI 682/670

A large part of the course focuses on **3D visual data analysis & synthesis (i.e., generative models)**



Kalogerakis et al, A probabilistic model for component-based shape synthesis, SIGGRAPH 2012

What is the course going to cover

Combines computer graphics, vision, and deep learning!

Tentative schedule:

Week 1-2: Image, Shape, Texture Representations [Assignment 1]

Week 3: Introduction to Neural Networks

Week 4: 3D Deep Learning: the multi-view approach [Assignment 2]

Week 5: 3D Deep Learning: the voxel-based & point-based approach

Week 6: 3D Deep Learning: the mesh-based approach [Assignment 3]

Week 7-8: Introduction to Generative Models cont'd

Week 9: 3D generative models: generating views, voxels, points [Assignment 4]

Week 10: 3D generative models: generating implicits and primitives

Week 11: 3D generative models: generating meshes and scenes [Assignment 5]

Week 12: Neural rigging and animation

Week 13: Neural Rendering

Course web page

Syllabus and general information:
http://people.cs.umass.edu/~kalo/courses/visual_computing/index.html

Moodle (lecture notes, assignments):
https://umass.moonami.com/course/view.php?id=35974

Piazza forum and Q&A:
https://piazza.com/umass/spring2023/cmpsci574674

Zoom (same for office hours & attending remotely):
https://umass-amherst.zoom.us/j/97550717142

Teaching plan

The same lectures are given in-person (except for any emergencies) **and transmitted online in real-time**.

Lectures are recorded and are made available on Moodle later.

You have the following options:

- Attend the Tue-Thu live meetings **in-person or remotely**
- **Watch the recorded lectures offline** and do not attend

Marking Scheme (**574**)

- **10 pts Assignment 1** (warm-up): Implicit shape representations
- **10 pts Assignment 2:** Multi-View Convolutional Networks
- **25 pts Assignment 3:** Point-based Networks
- **20 pts Assignment 4:** Image Synthesis
- **20 pts Assignment 5:** Reconstruction
- **15 pts Research Paper presentation (recorded video)**

Assignments and paper presentation are done individually!!!

Paper Presentations (574)

I will post a list of papers towards the **middle of the semester.** You will select one from the list.

You may also choose one on your own, but I have to approve it first.

~15 min **pre-recorded presentation**

I will select the best ~5 presentations to play them in-class *probably* during the last week (+these will get a small bonus)

Assignments

Read course policy on Moodle!!!

- Programming assignments will be on **Python**
- Goal: algorithms should produce **correct results** and **be reasonably fast.**
- First **warm-up** assignment released **on Thu**
- You have **2 weeks** to complete assignments
- **Late policy: -20%** of the final mark for every day you delay the submission. **Plagiarism gives you 'F'.**

Marking Scheme (**674**)

- **7.5 pts Assignment 1** (warm-up): Implicit shape representations
- **7.5 pts Assignment 2:** Multi-View Convolutional Networks
- **20 pts Assignment 3:** Point-based Networks
- **15 pts Assignment 4:** Image Synthesis
- **15 pts Assignment 5:** Reconstruction
- **10 pts Project Proposal**
- **25 pts Final Project**

Assignments are done individually! (same with 574)

Project is done in teams 2-3 people!

Project (674)

Small-scale research project

~15 min **presentation.**

I will select the best ~5 presentations to play them in-class *probably* during the last week (+these will get a small bonus)

Topic examples [must be related to 3D/4D deep learning]:

- *generative models of shapes/scenes*
- *3D shape/scene segmentation*
- *surface reconstruction*
- *neural rendering*
- *neural animation*
- *3D style transfer*

Project (674)

Computing Resources (same as 682):

Amazon offers free AWS credits for students (annually renewable). Using your link provided by the GitHub Student Developer Pack will get you the most free credits. UMass is an "AWS member institution", so you are in the higher allowance tier. Use your .edu email and the full school name "University of Massachusetts Amherst" when you register to get the full benefits (a total of $75-$150).

To get GPUs, use g3 (up to 4 NVIDIA Tesla M60 GPUs) or p2 (up to 16 NVIDIA K80 GPUs) instances in EC2. Check the pricing first and make your plan accordingly!

Google offers $300 google cloud credits if you sign up for the first time.
See the Google Cloud Tutorial.

Academic Dishonesty

If you copy **code** from other students (including from previous years) OR online sources, you get an “F”.

Excuses are not accepted:

- “I did not understand that looking at solutions online/from others, was considered cheating.”
- “I did not understand that working with other students on solutions, was considered cheating.”
- “I am taking too many courses / I am desperate.”
- “I thought that the penalty would not be severe.”

Academic Dishonesty

If you can't handle an assignment/project, consider the following options **instead of cheating:**

- Withdraw (it's better than "F"!).
  [Note: if you cheat, withdrawing is not an option anymore]
- It's ok to lose points in an assignment. Doing a great project can give you a BONUS to recover lost marks.
- Get help from the TAs
- Obtain accommodations:
  https://www.umass.edu/disability/students/accommodations-students

Instructor & TAs

TA office hours: Fri 10am - 12pm ET

Instructor office hours: Mon 11:30-12pm & 1:30-2:30pm ET

TAs: Fabien Delattre, Dmitry Petrov, Vikas Thamizharasan

Don’t email me or TAs - use piazza (private posts are OK)

Instructor & TA office hours are REMOTE (same link as class)

Tips to succeed

Don’t cheat!

Watch the classes!

Start early on each assignment and project!

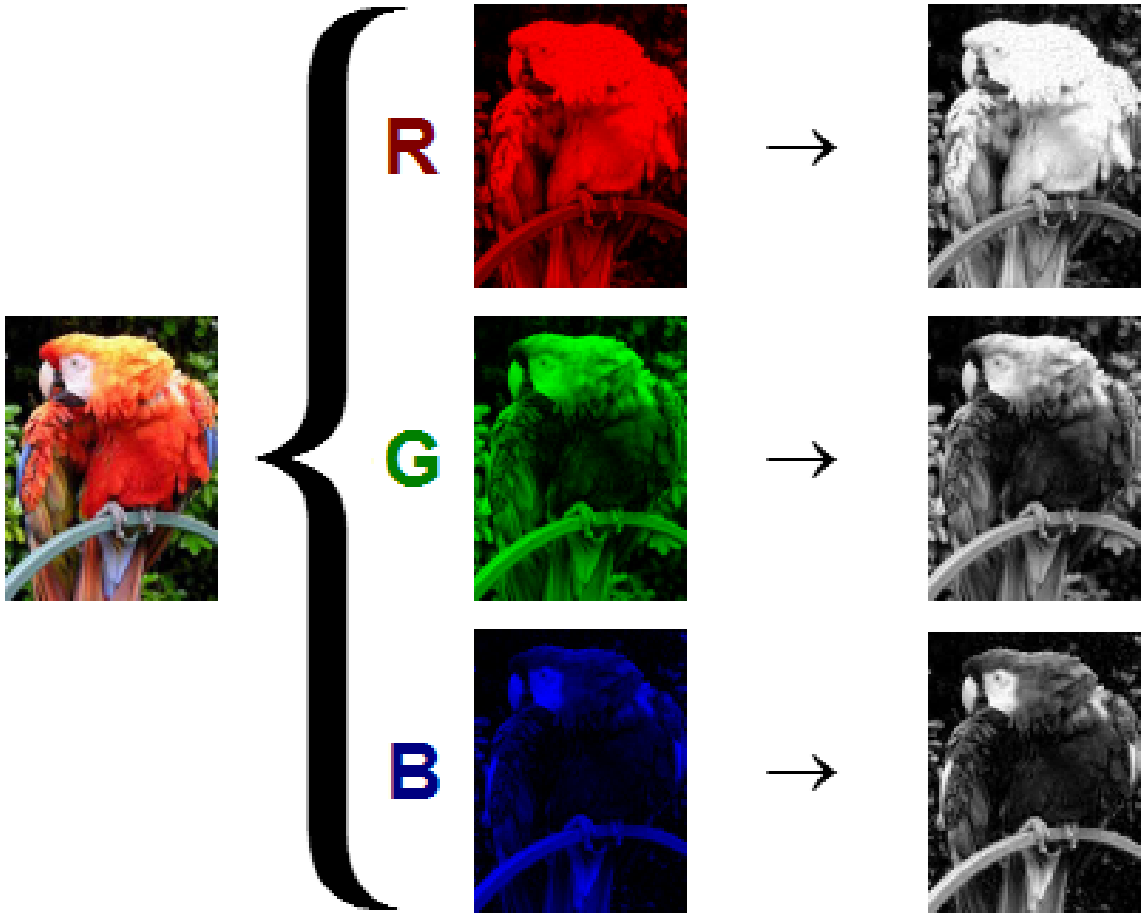
Work hard!

Introduction to image and shape representations

Part I: Image representations

How do we represent images?

Images are often represented as arrays of pixels with **RGB** color.



How do we represent images?

We can think of an **image** as a function, f, from R^2 to R:

- $f(x, y)$ gives the **intensity** at position (x, y)
- Realistically, we expect the image only to be defined over a rectangle, with a finite range:
  - $f: [a,b]\mathrm{x}[c,d] \rightarrow [0,1]$

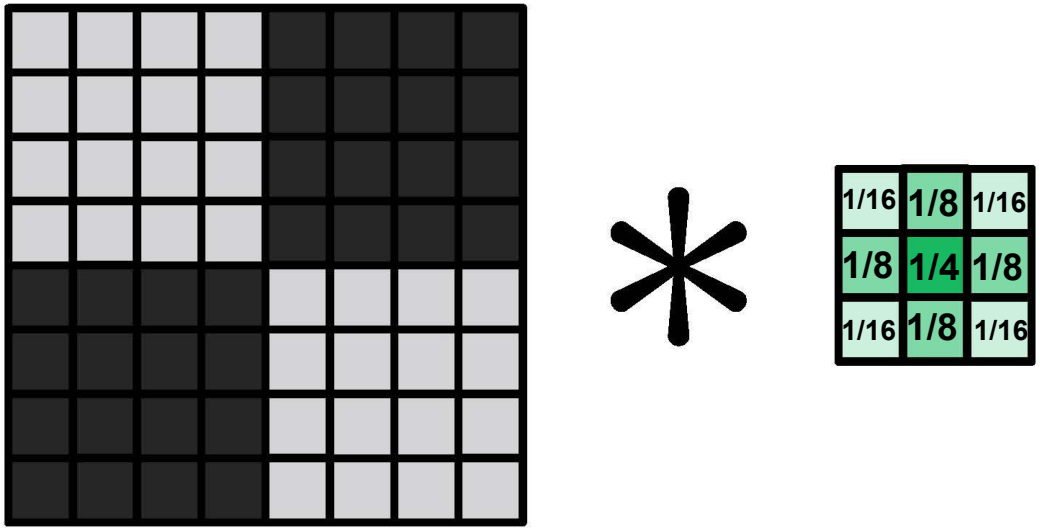
A color image is just three functions pasted together. We can write this as a "vector-valued" function:

$$f(x,y) = \begin{bmatrix} red(x,y) \\ green(x,y) \\ blue(x,y) \end{bmatrix}$$

Review: 2D Convolution

$$O(i,j) = \sum_{k=-n}^{k=n} \sum_{l=-n}^{l=n} w(k,l) I(i+k, j+l)$$

Note: formally, this is called cross-correlation.

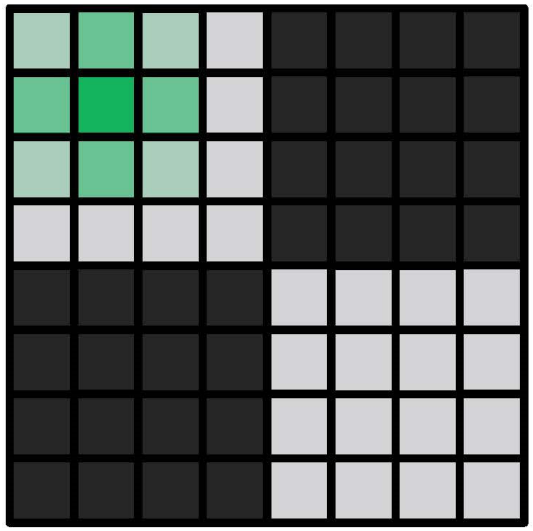


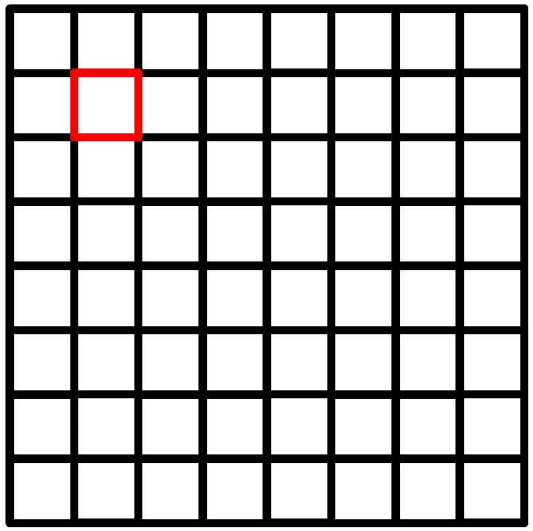
$$O(i,j) = \sum_{k=-n}^{k=n} \sum_{l=-n}^{l=n} w(k,l) I(i-k, j-l)$$

Note: formally, this is convolution. The minus difference (i.e., flipping) is not important for neural networks since the filters are learned.

2D Convolution Review

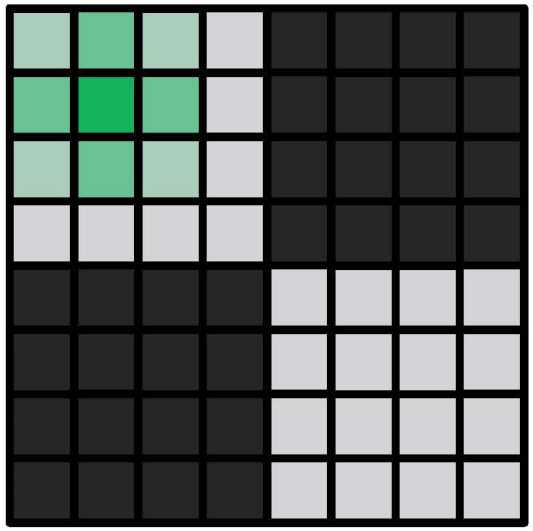
$$O(i,j) = \sum_{k=-n}^{k=n} \sum_{l=-n}^{l=n} w(k,l) I(i+k, j+l)$$

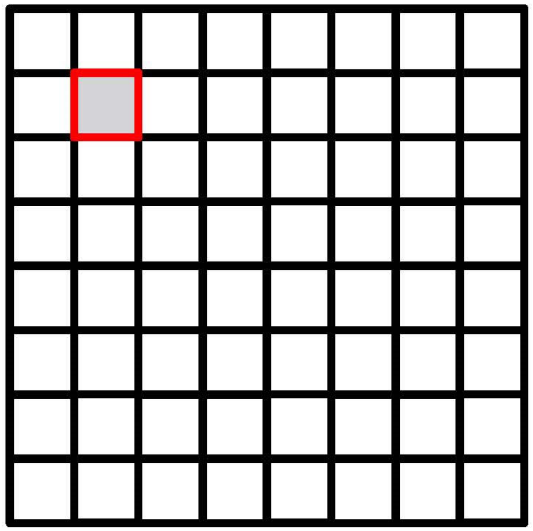




2D Convolution Review

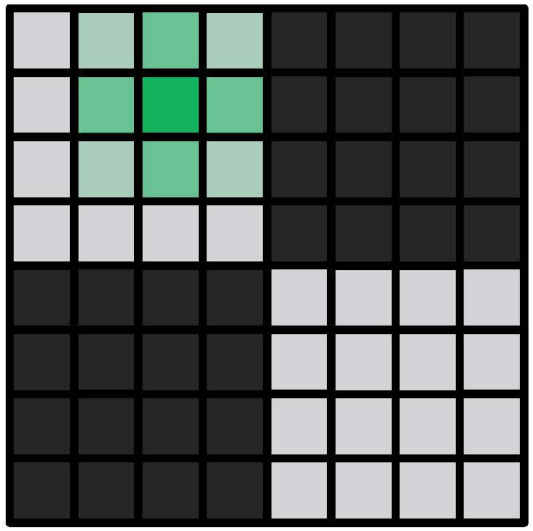
$$O(i,j) = \sum_{k=-n}^{k=n} \sum_{l=-n}^{l=n} w(k,l) I(i+k, j+l)$$

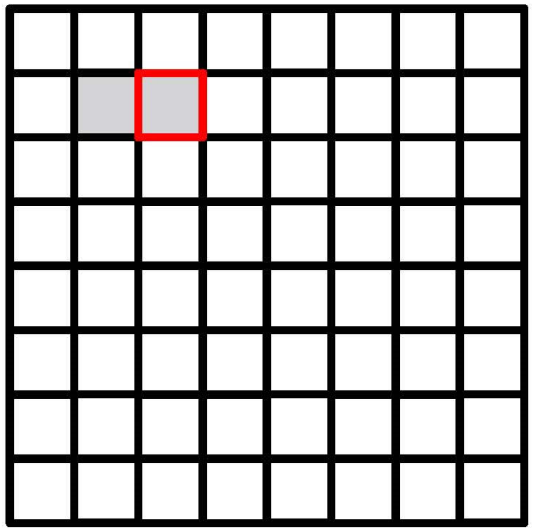




2D Convolution Review

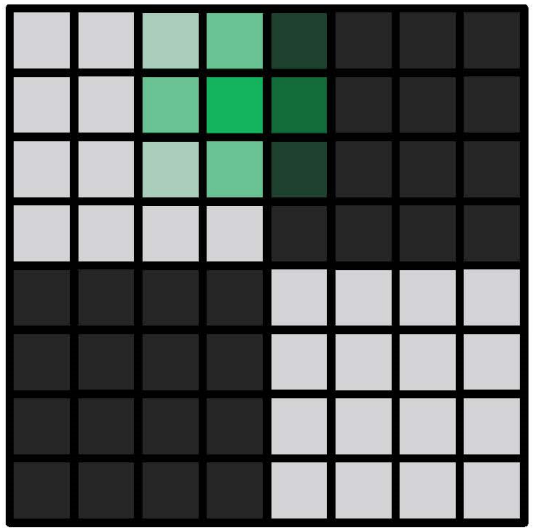
$$O(i,j) = \sum_{k=-n}^{k=n} \sum_{l=-n}^{l=n} w(k,l) I(i+k, j+l)$$

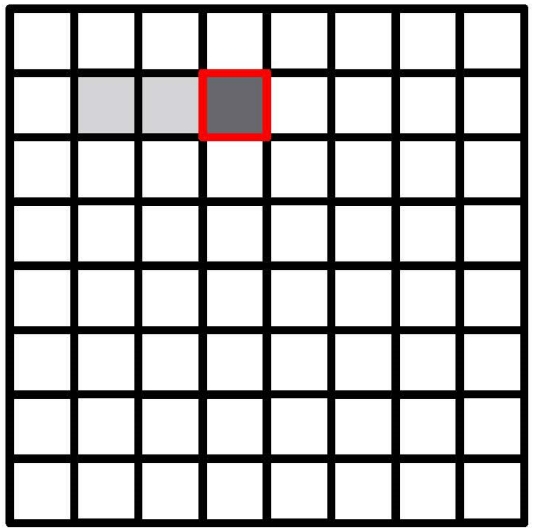




2D Convolution Review

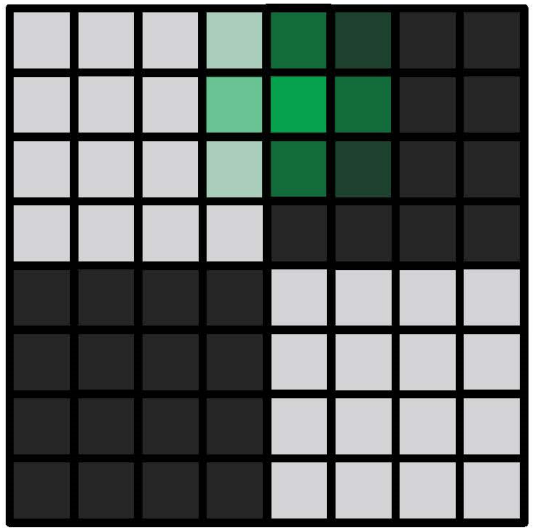
$$O(i,j) = \sum_{k=-n}^{k=n} \sum_{l=-n}^{l=n} w(k,l) I(i+k, j+l)$$

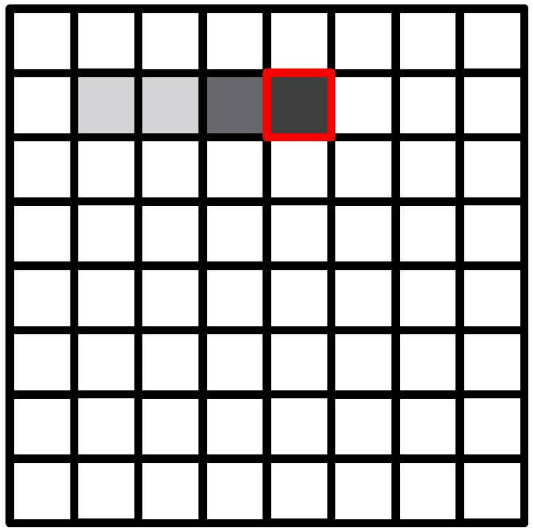




2D Convolution Review

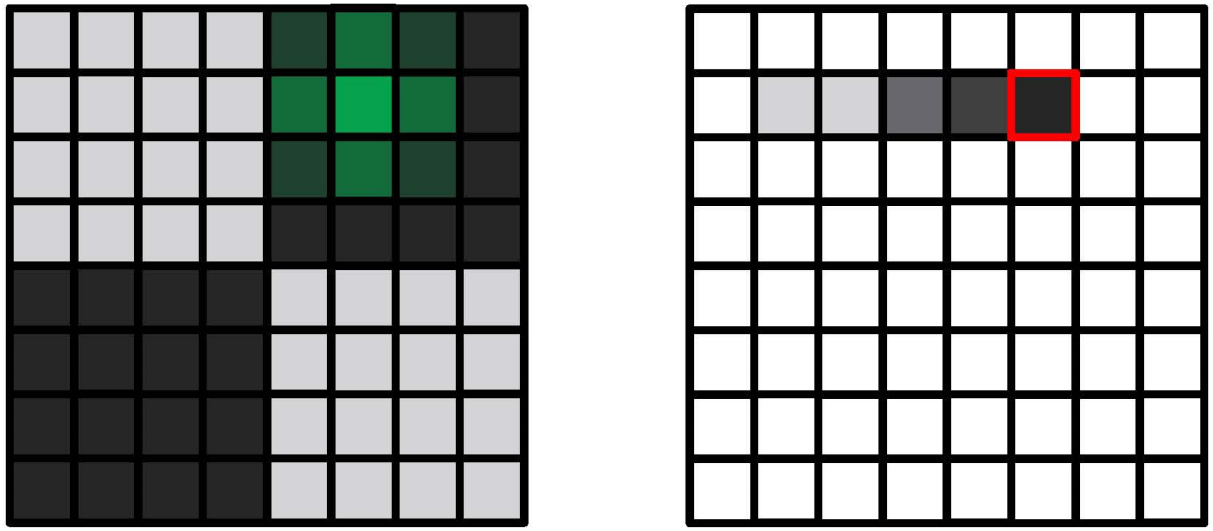
$$O(i,j) = \sum_{k=-n}^{k=n} \sum_{l=-n}^{l=n} w(k,l) I(i+k, j+l)$$





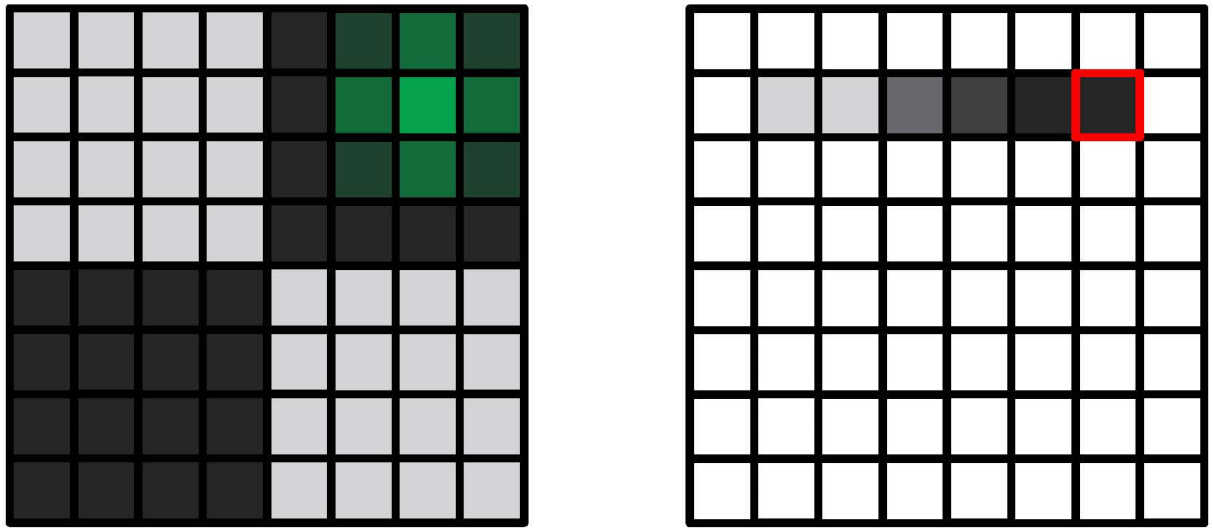
2D Convolution Review

$$O(i,j) = \sum_{k=-n}^{k=n} \sum_{l=-n}^{l=n} w(k,l) I(i+k, j+l)$$



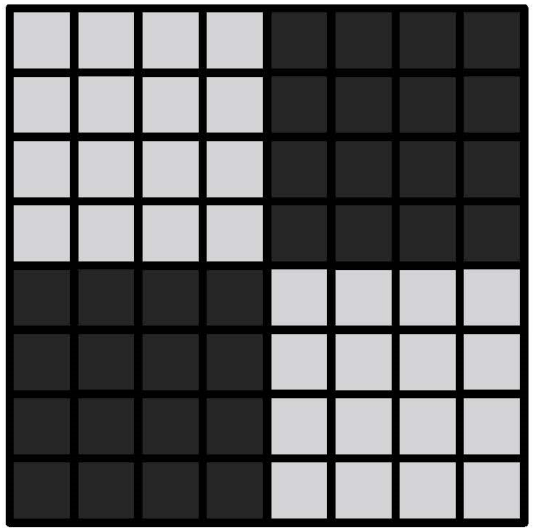
2D Convolution Review

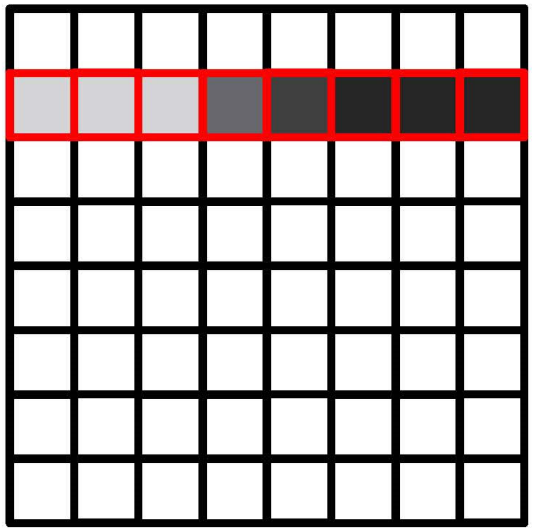
$$O(i,j) = \sum_{k=-n}^{k=n} \sum_{l=-n}^{l=n} w(k,l) I(i+k, j+l)$$



2D Convolution Review

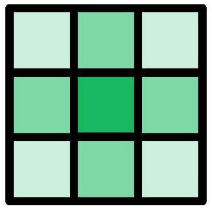
$$O(i,j) = \sum_{k=-n}^{k=n} \sum_{l=-n}^{l=n} w(k,l) I(i+k, j+l)$$



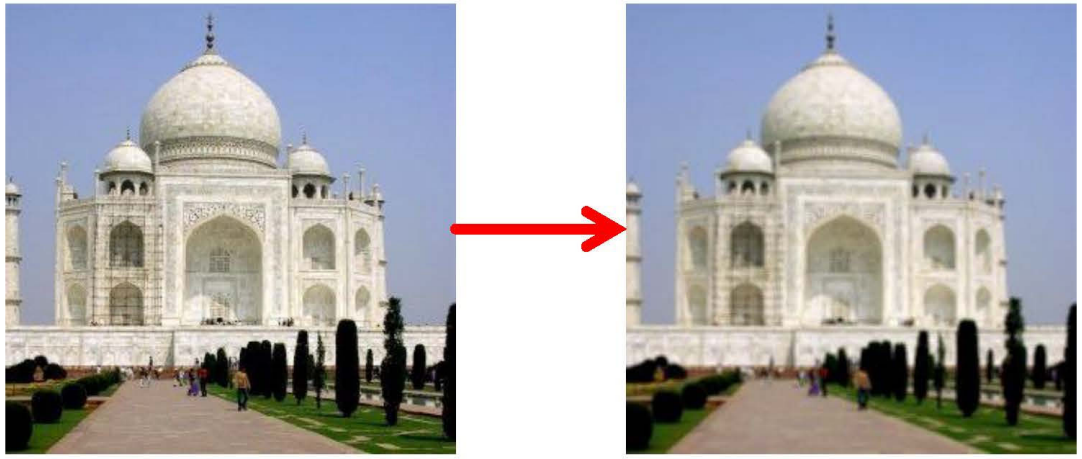


2D Convolution Review (Blurring)

$$\begin{bmatrix} 1/16 & 1/8 & 1/16 \\ 1/8 & 1/4 & 1/8 \\ 1/16 & 1/8 & 1/16 \end{bmatrix}$$



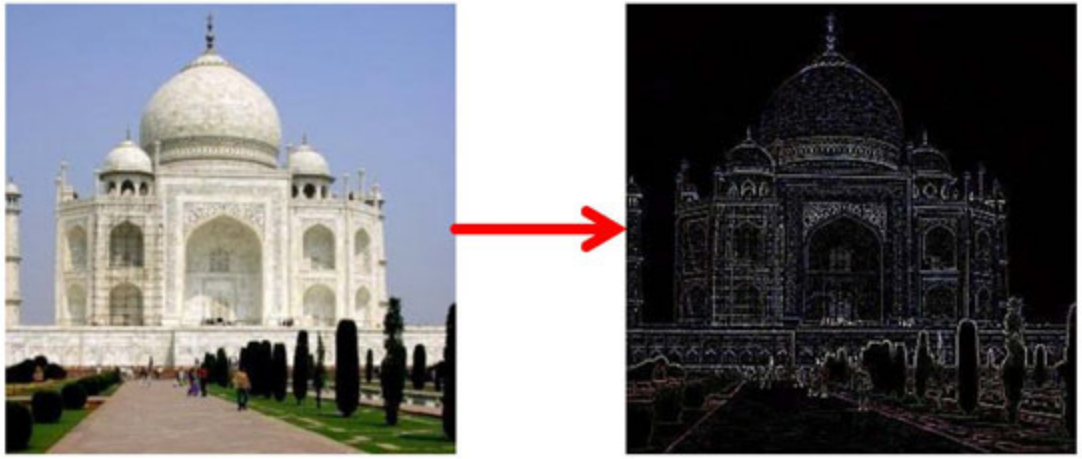
Blur



2D Convolution Review (Edge Detection)

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

Edge detect



Introduction to image and shape representations

Part II: Shape representations

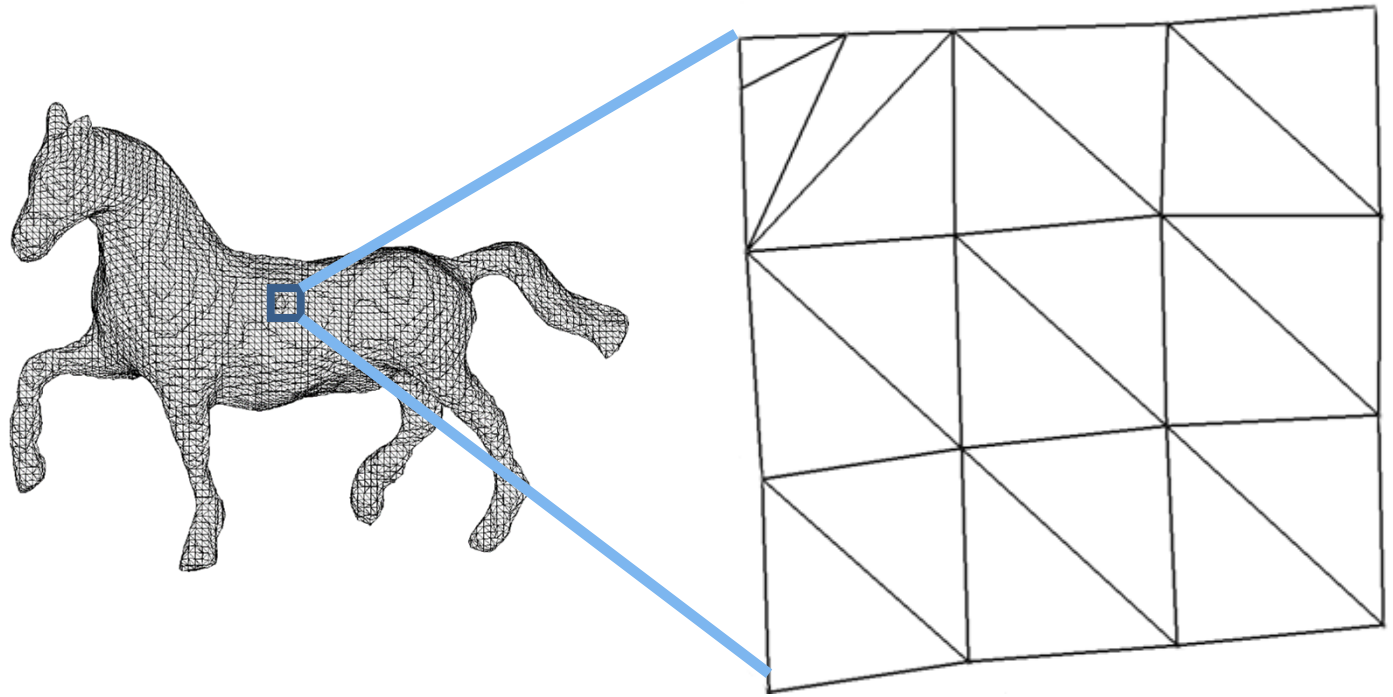
Part II: Shape representations

- **Polygon Meshes**
- Parametric Surfaces
- Voxel grids
- Point clouds
- Implicit functions

Polygon Meshes

Most common digital shape representation in graphics

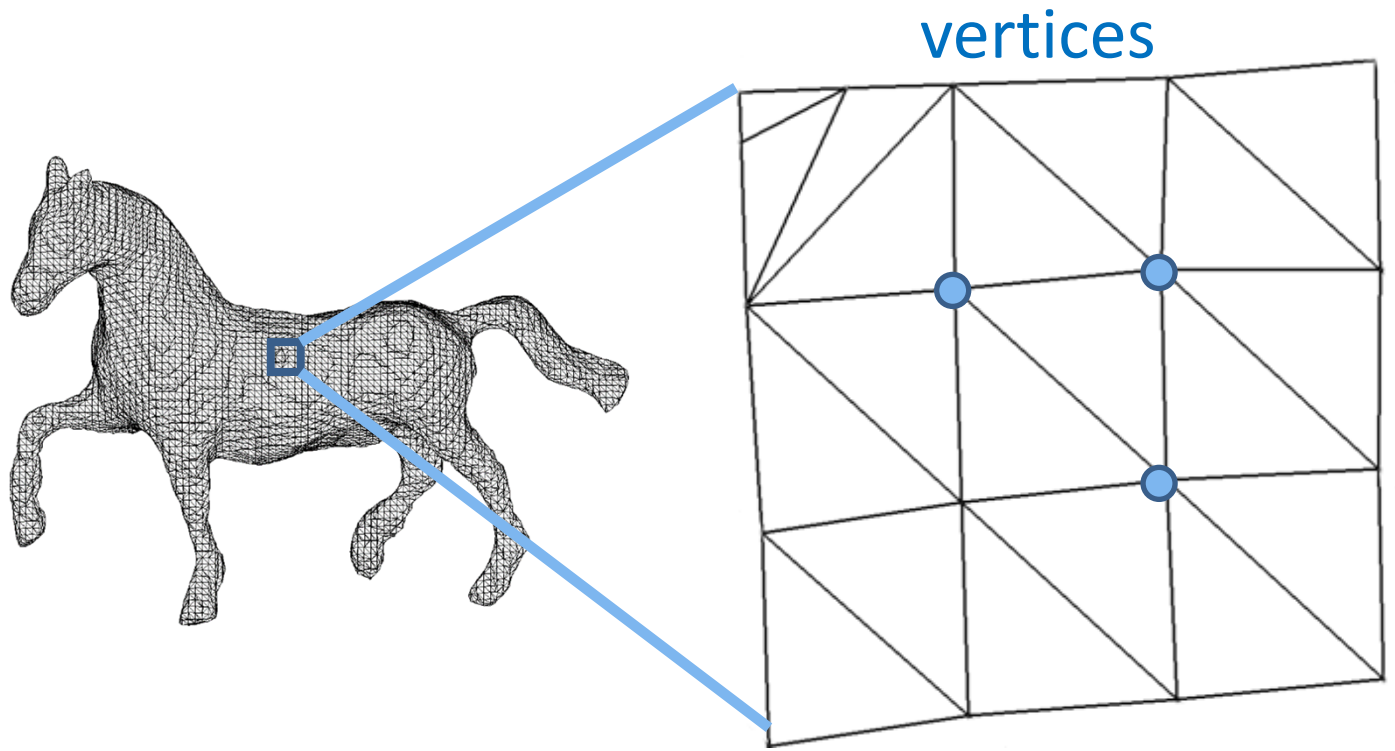
Collection of polygons (faces) that form the “skin” of an object



Polygon Meshes

Most common digital shape representation in graphics

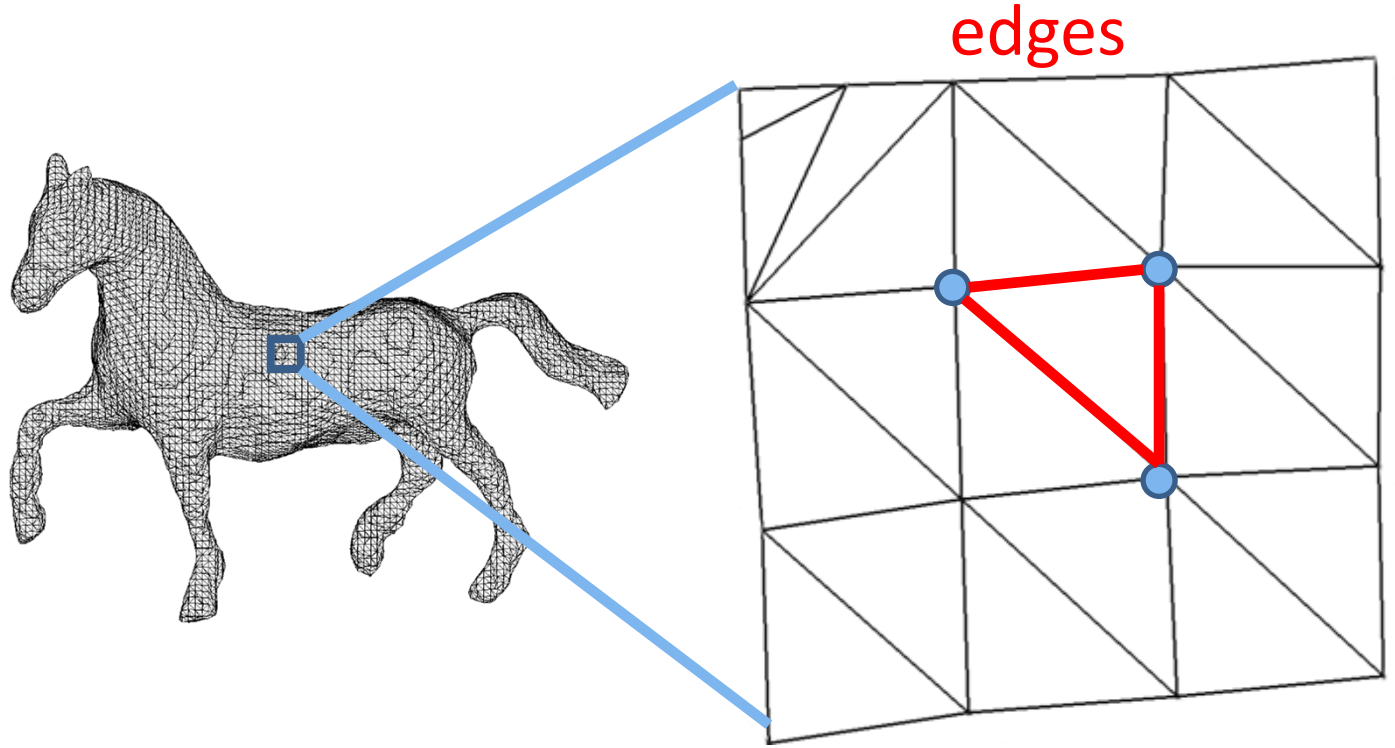
Collection of polygons (faces) that form the “skin” of an object



Polygon Meshes

Most common digital shape representation in graphics

Collection of polygons (faces) that form the “skin” of an object



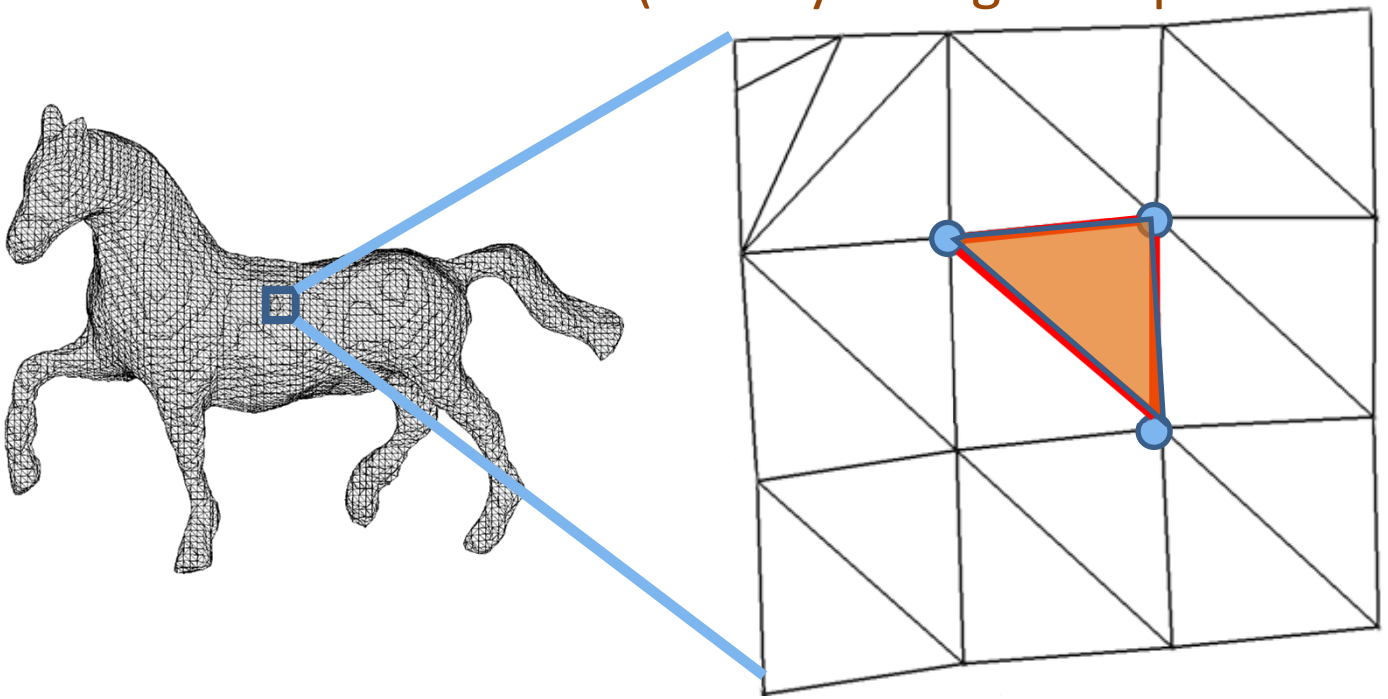
Polygon Meshes

Most common digital shape representation in graphics

Collection of polygons (faces) that form the “skin” of an object

face

(usually triangle or quadrilaterals)



Data Structures for meshes

- What should we store?
  - Geometry: 3D coordinates of polygon vertices
  - Other mesh properties e.g. connectivity information, normals, colors, texture coordinates, etc.

- Common mesh representation: **vertex and face list**

OBJ file format

Comments start with '#'
List of Vertices, with (x,y,z) coordinates
v 0.123 0.234 0.345

v 0.433 0.348 0.438

…

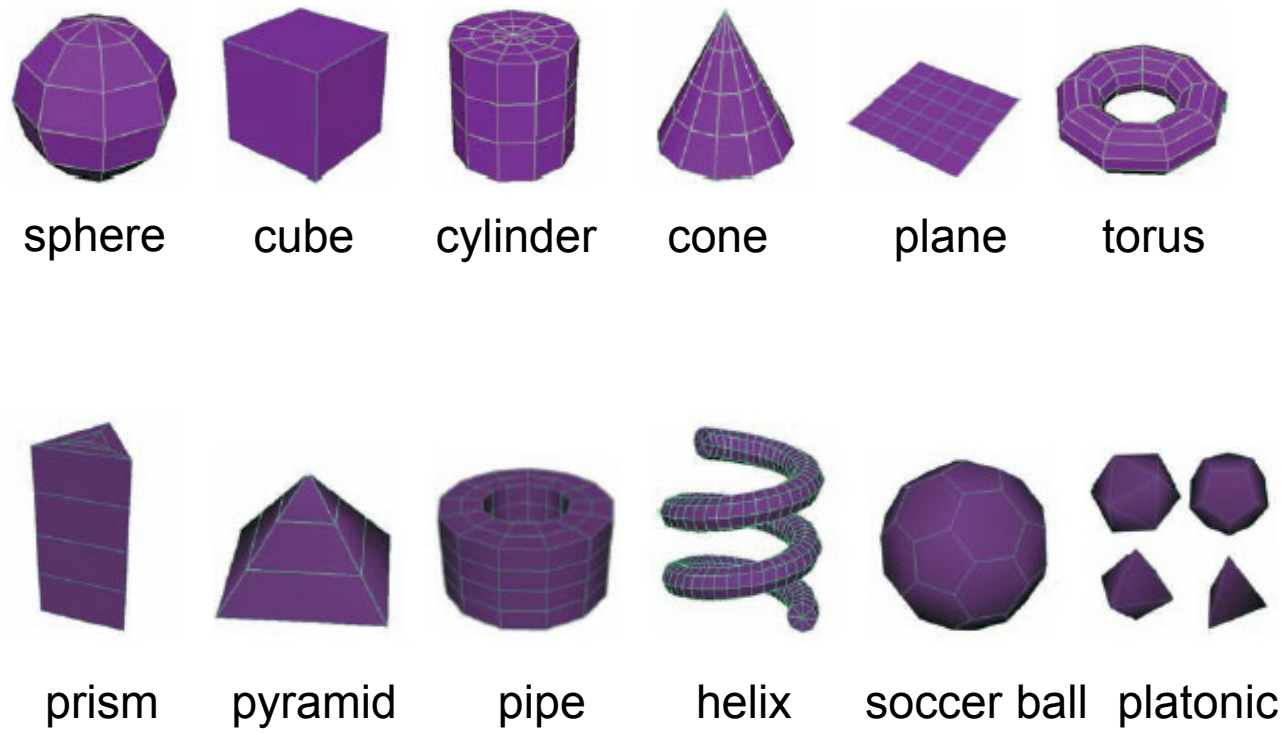
Then list of faces (**vertex index starts from 1**)
f 1 2 3

f 3 4 5

…

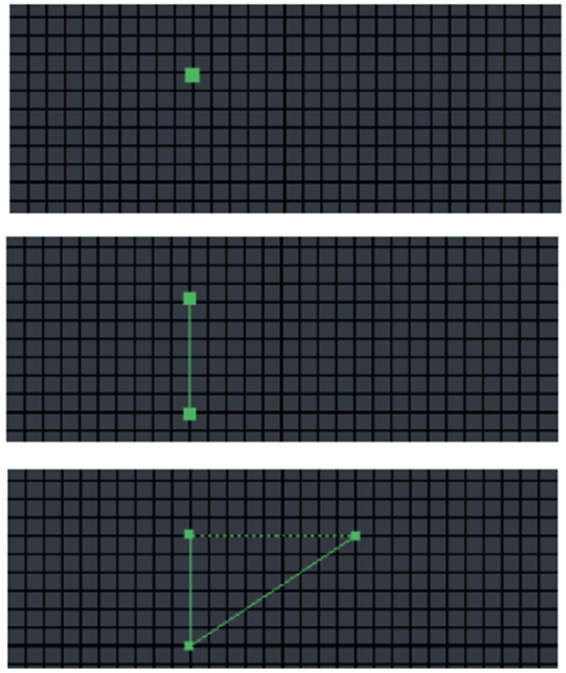
Polygonal modeling

Start with predefined primitives...



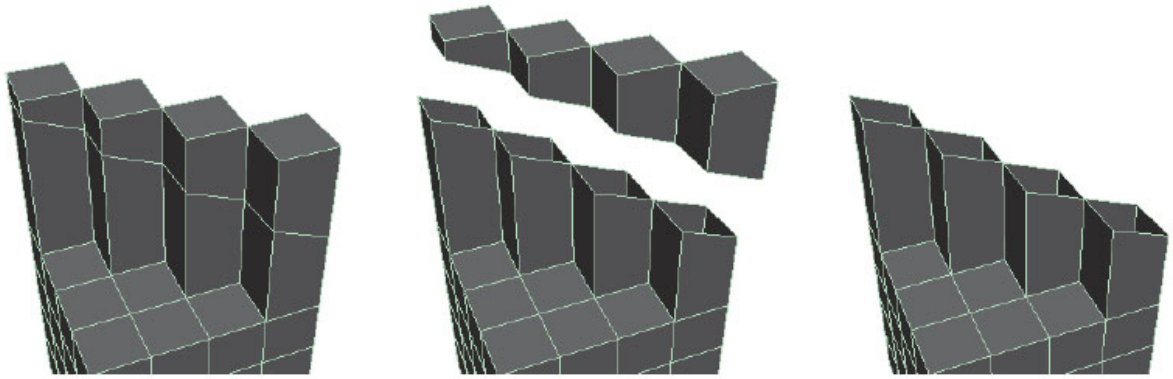
Polygonal modeling

Click and add/delete/move vertices...



Polygonal modeling

Specify cutting line / plane to cut polygon mesh



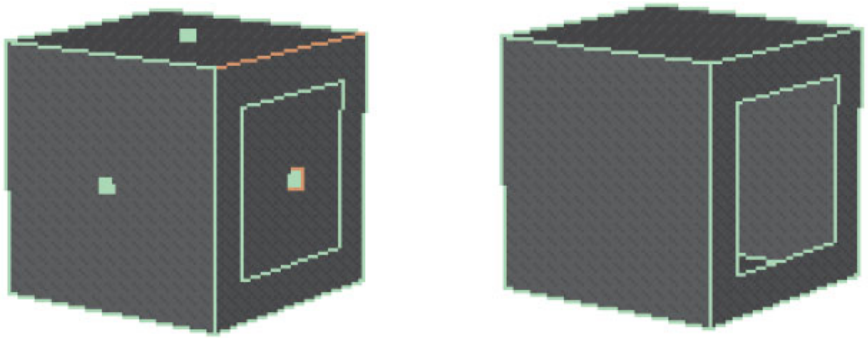
Use Cut Faces to create edges...

...to extract faces, or...

...to delete faces.

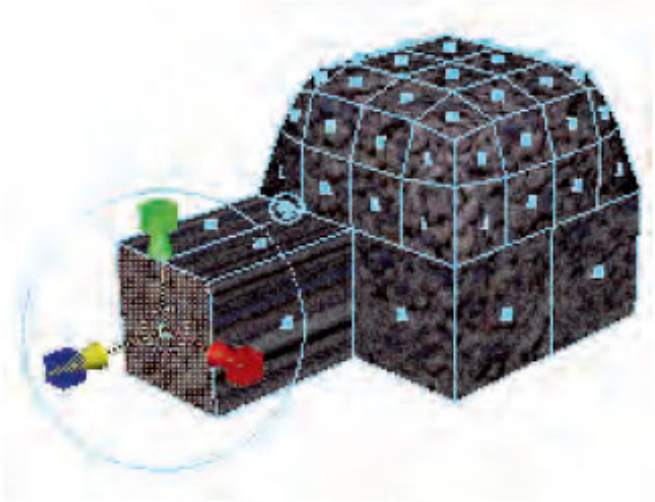
Polygonal modeling

Specify a polygon to create a hole in another polygon

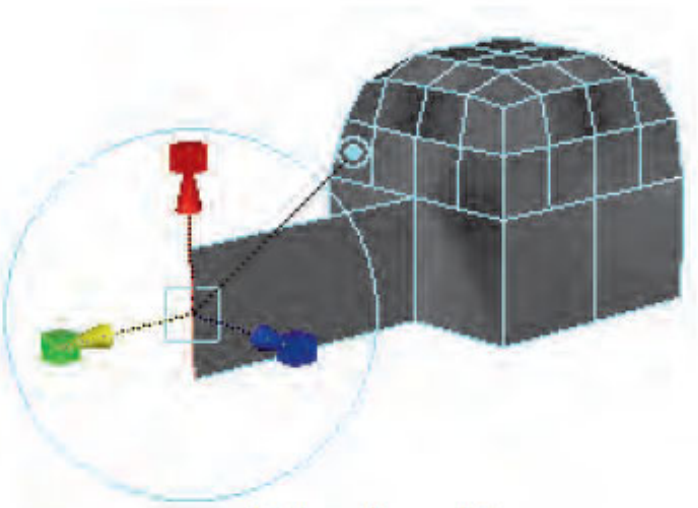


Polygonal modeling

Extrude polygon faces, edges or vertices along some specified direction

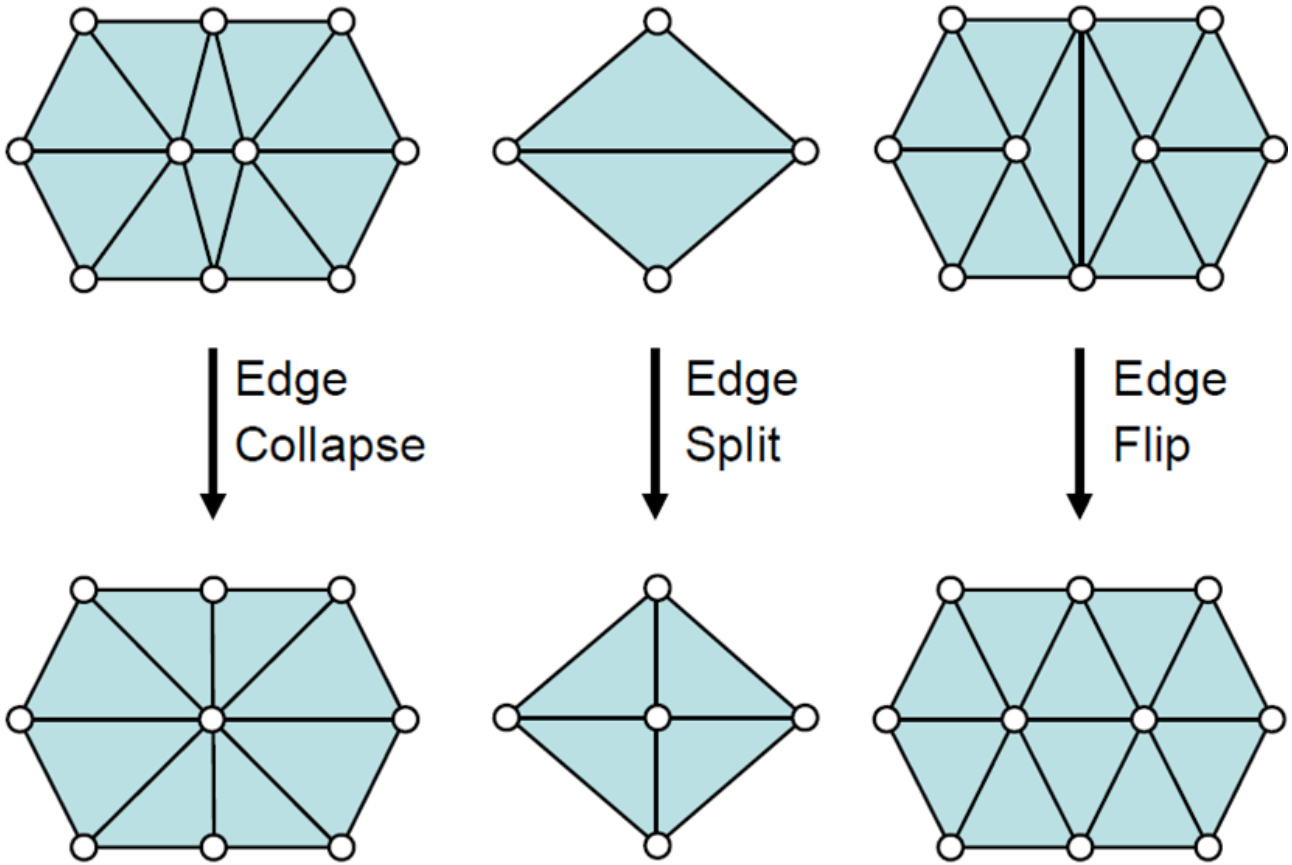


Extrude face

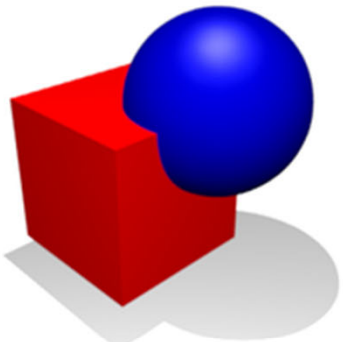


Extrude edge

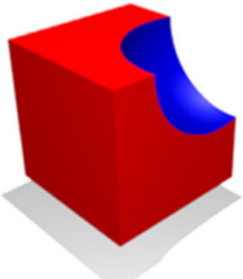
Polygonal modeling



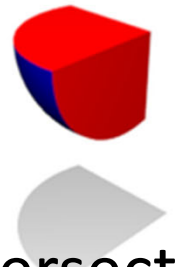
Boolean Operations (Constructive Solid Geometry)



A union **B**

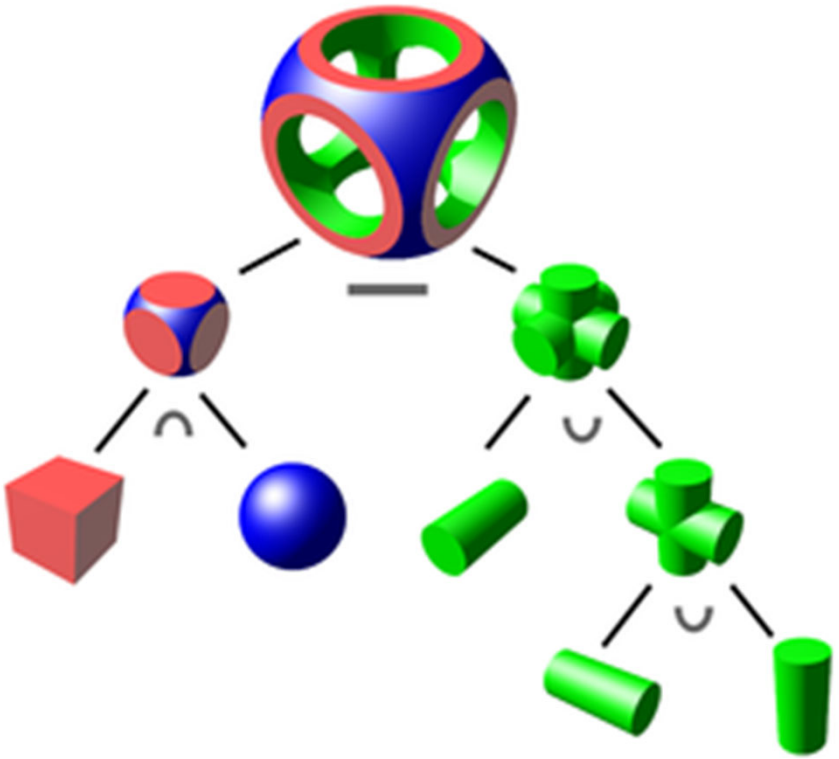


A diff **B**

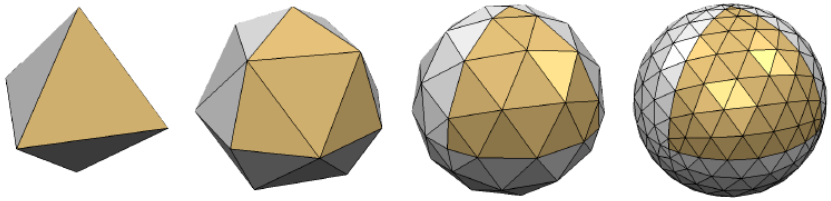


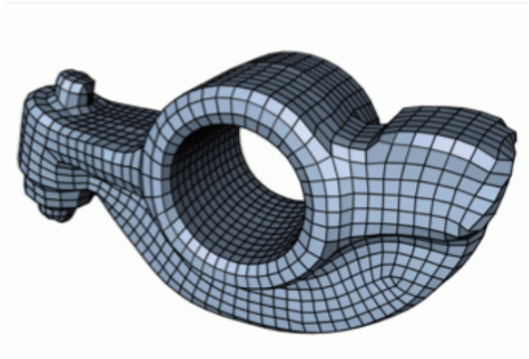
A intersection **B**

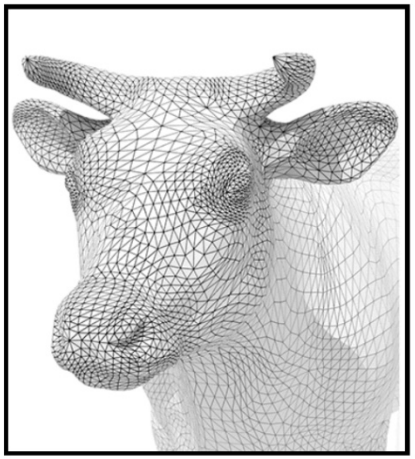
Boolean Operations (Constructive Solid Geometry)

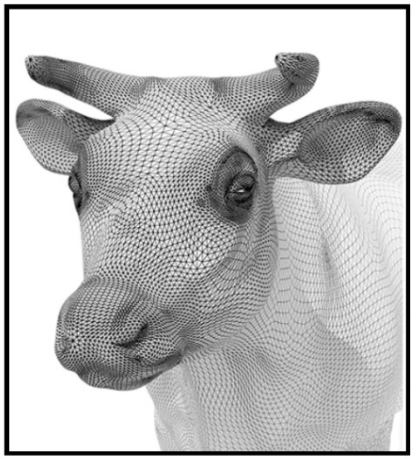


Polygon Mesh
Advantages/Disadvantages









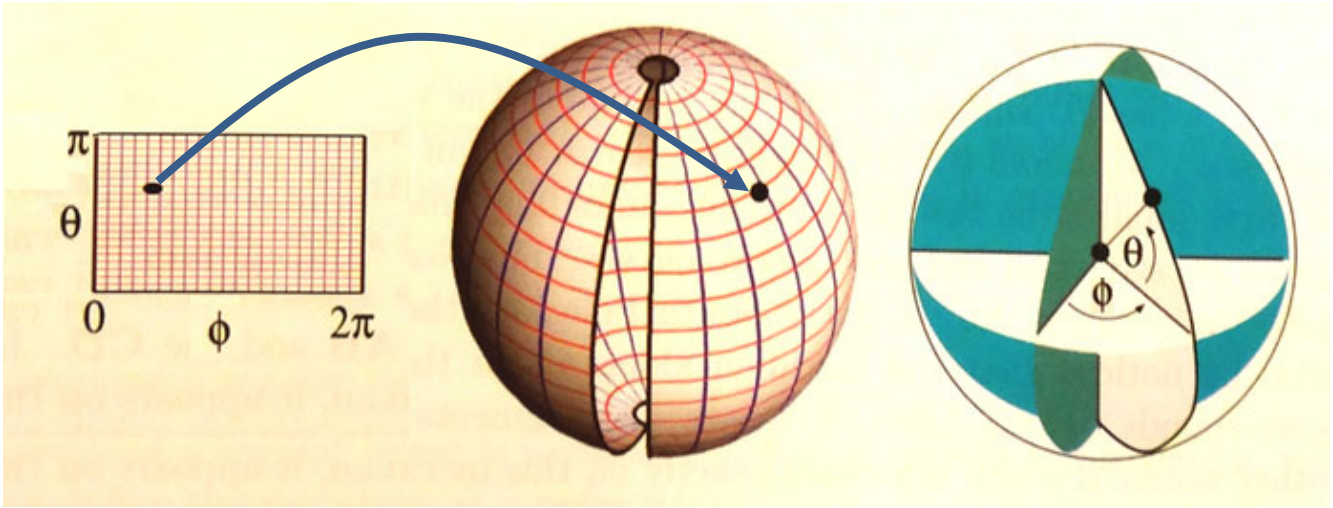
- **Rendering is plausible and fast**
- **Many techniques for mesh deformation, animation, subdivision...**
- **Unordered representation: vertex order is arbitrary, no fixed number of vertices, different #neighbors per vertex...**

Part II: Shape representations

- Polygon Meshes
- **Parametric Surfaces**
- Voxel grids
- Point clouds
- Implicit functions

Parametric Representation of Surfaces

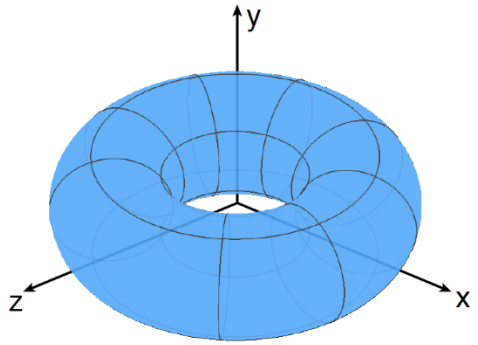
A smooth surface in 3D can be thought of as a map from a 2D set of parameters to 3D space



$$x(\theta,\varphi) = R\sin\theta\cos\varphi$$
$$y(\theta,\varphi) = R\sin\theta\sin\varphi$$
$$z(\theta,\varphi) = R\cos\theta$$
$$\theta \in (0,\pi), \varphi \in (0,2\pi)$$

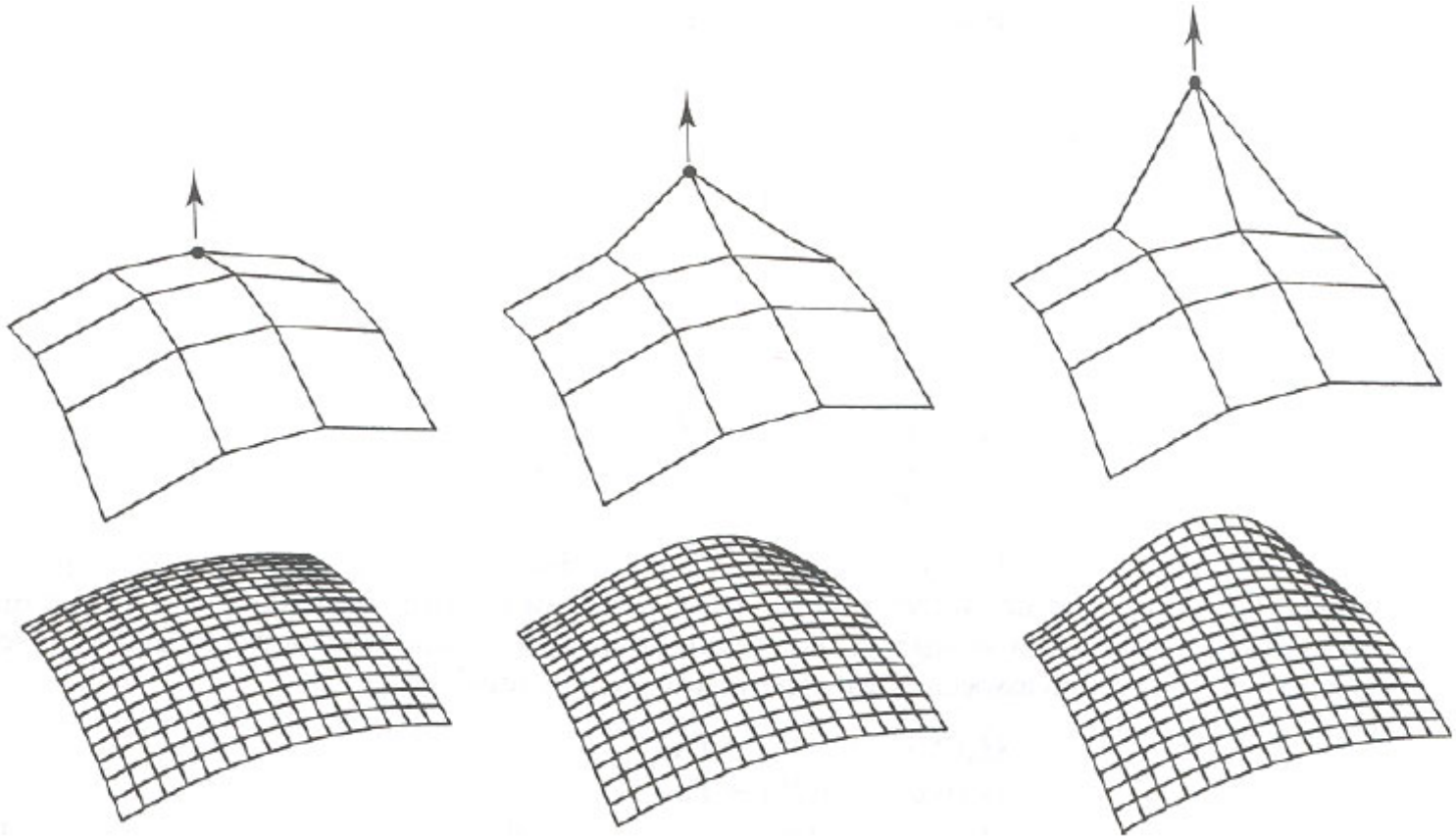
Parametric surface representation

A smooth surface in 3D can be thought of as a map from a 2D set of parameters to 3D space



$$f(u, v) = ((2 + \cos u) \cos v, (2 + \cos u) \sin v, \sin u)$$

Parametric patches

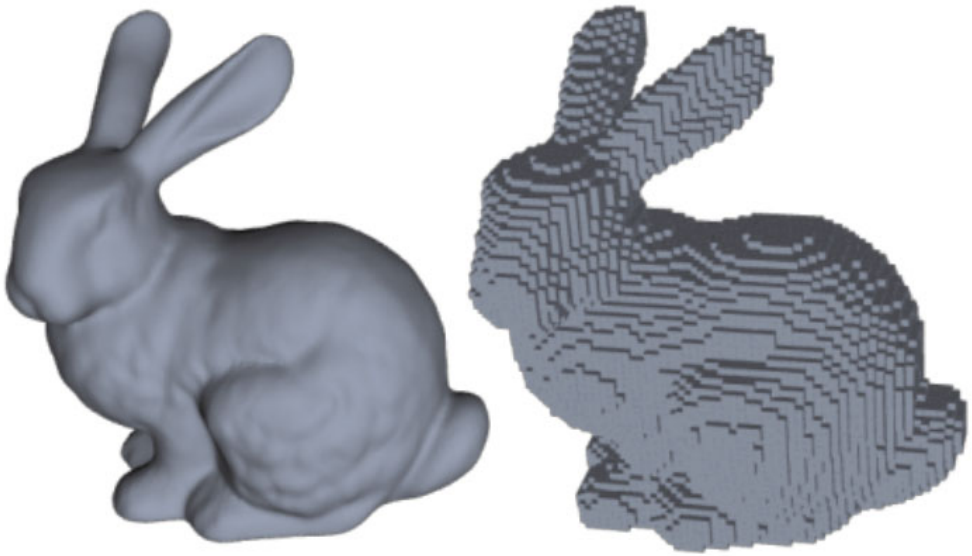


- **Surface is continuous**
- **Rendering is plausible**
- **Represents specific topologies**

Part II: Shape representations

- Polygon Meshes
- Parametric Surfaces
- **Voxel grids**
- Point clouds
- Implicit functions

Voxel representation



$$f(x,y,z)=\begin{cases}0 & if\ voxel\ is\ outside\ the\ shape\ (unoccupied)\\ 1 & if\ voxel\ is\ inside\ the\ shape\ /\ contains\ surface\ (occupied)\end{cases}$$

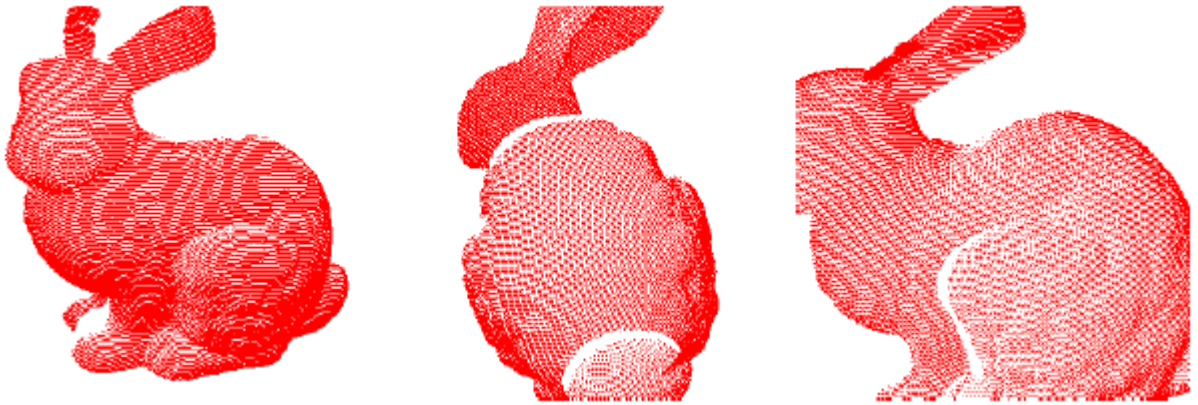
- **Ordered representation (representation in a NxNxN grid)**
- **Implausible to display in low-resolutions**
- **Expensive to store at high resolutions (N^3 voxels)**

Part II: Shape representations

- Polygon Meshes
- Parametric Surfaces
- Voxel grids
- **Point clouds**
- Implicit functions

Point Clouds

A collection of points without connectivity information...
usually the raw output of a scan



3D scanning

Acquire geometry of a physical object

Physical object

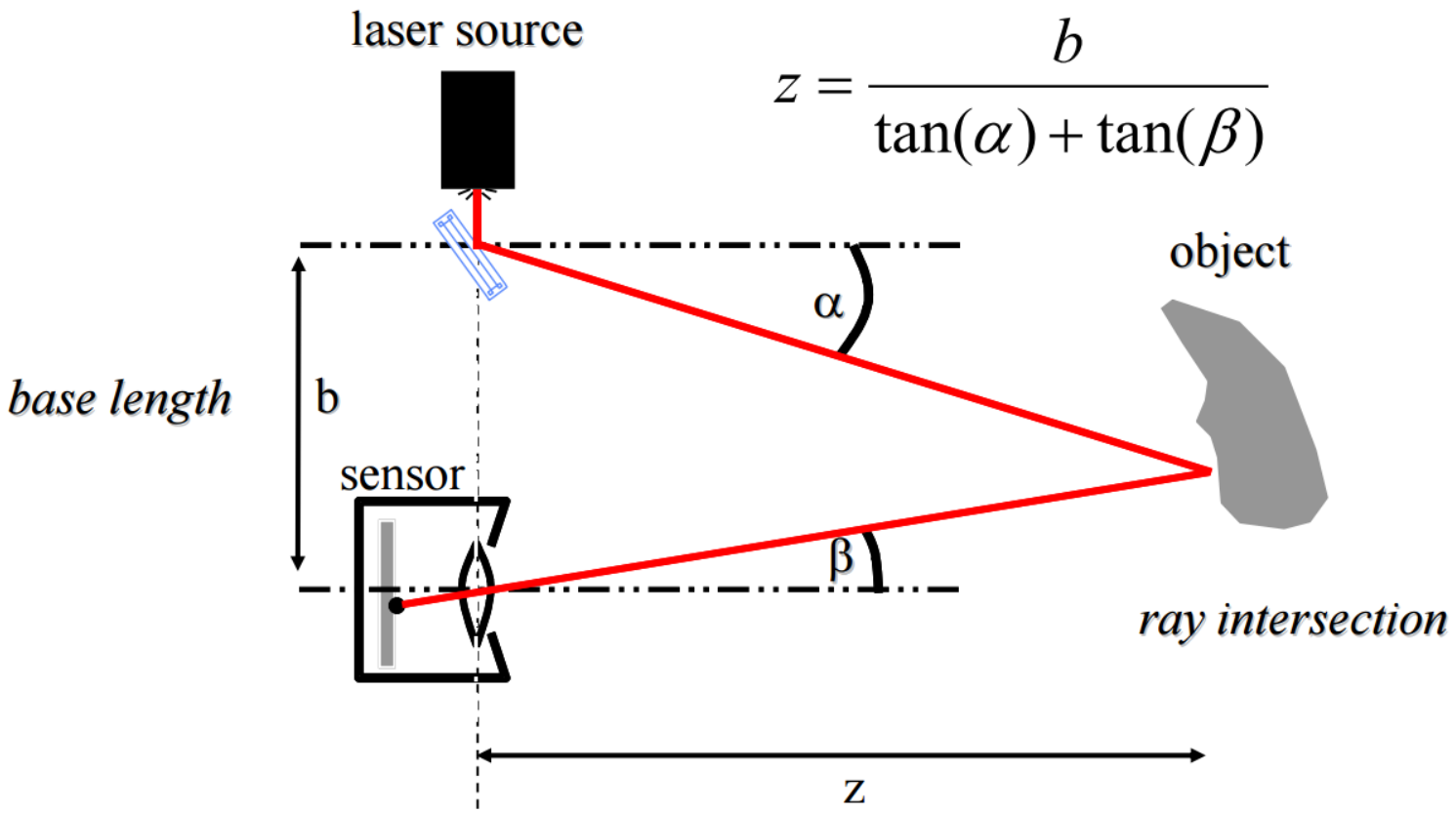
scanner

Raw scans

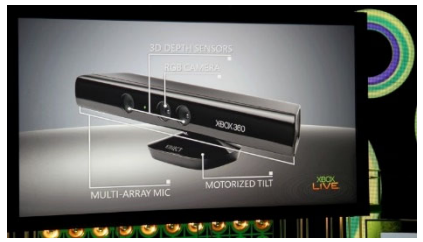
Basic Idea

(laser scanning)

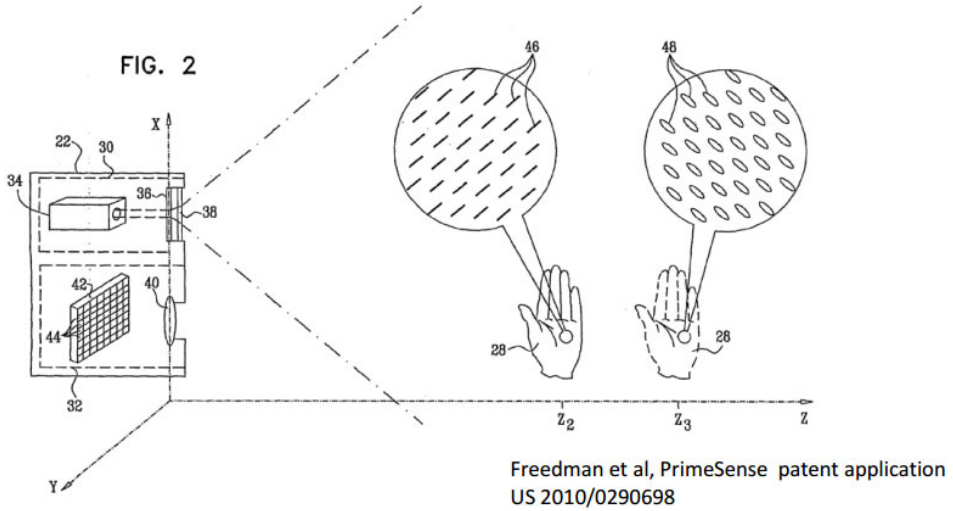
Project light onto an object, analyze reflected light



Basic Idea
(Structured light scanning)



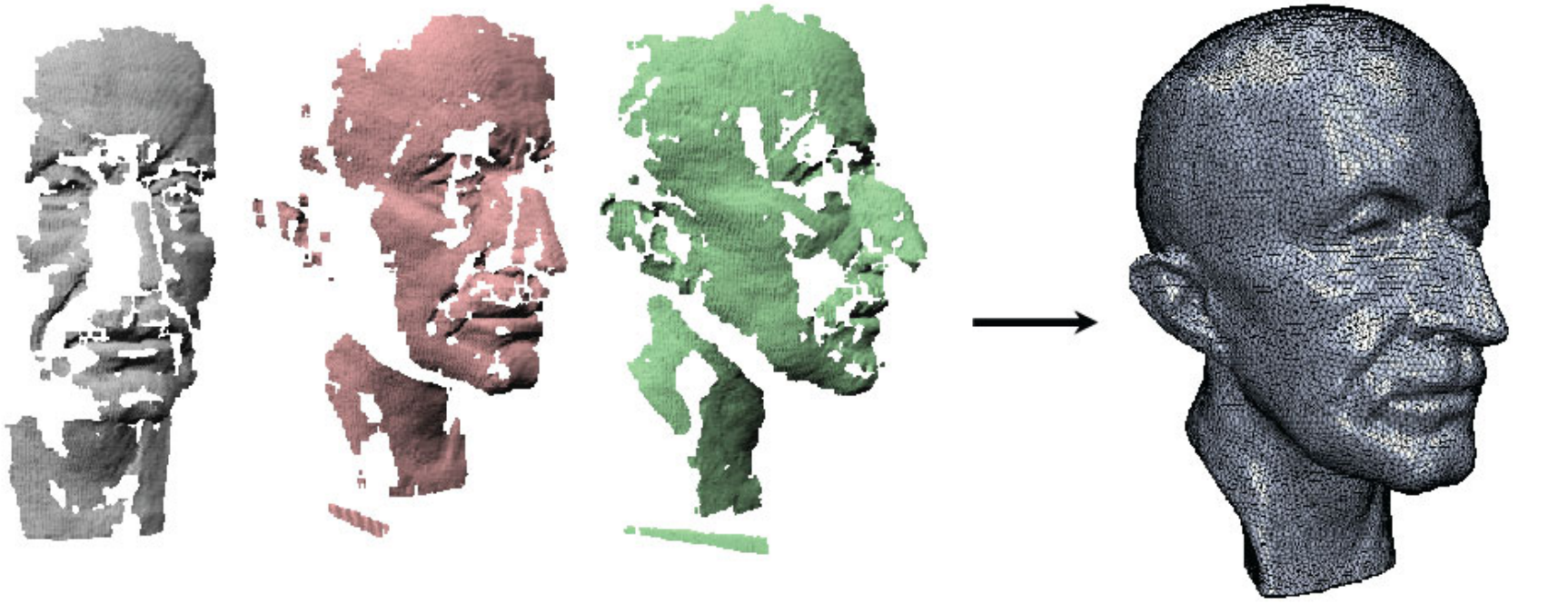
Kinect projects light patterns through a special lens ("astigmatic") with different focal length in x- & y-directions.



Freedman et al, PrimeSense patent application US 2010/0290698

A projected circle then becomes an ellipse. Measures depth based on deformation

From Point Clouds to Surfaces



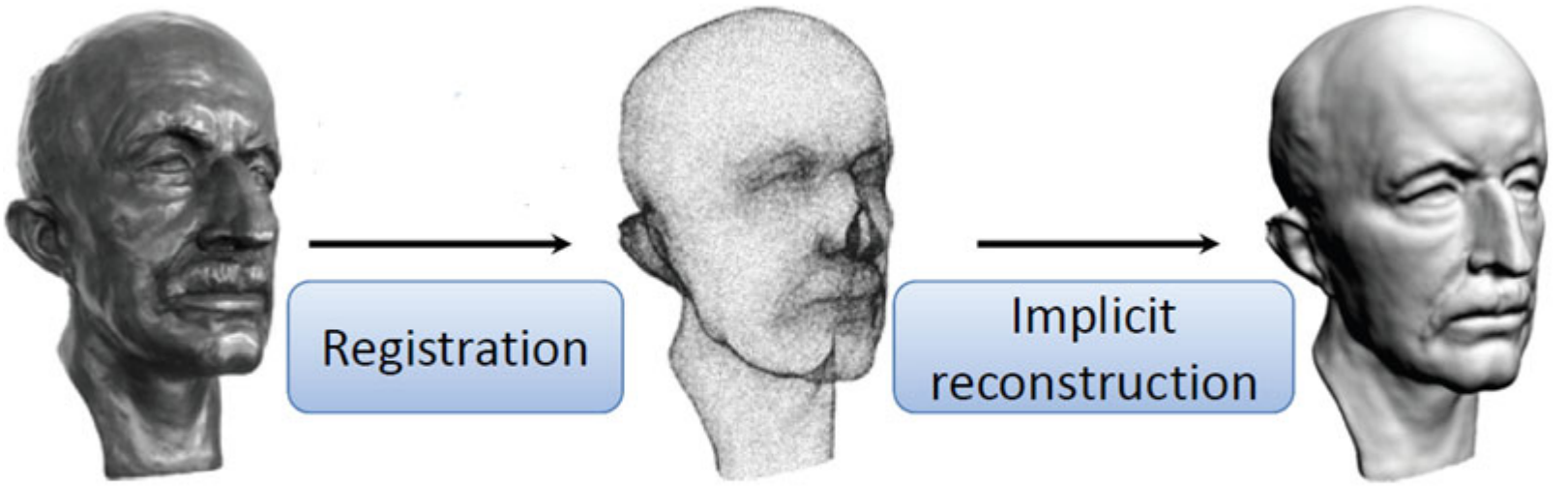
set of raw scans

reconstructed model

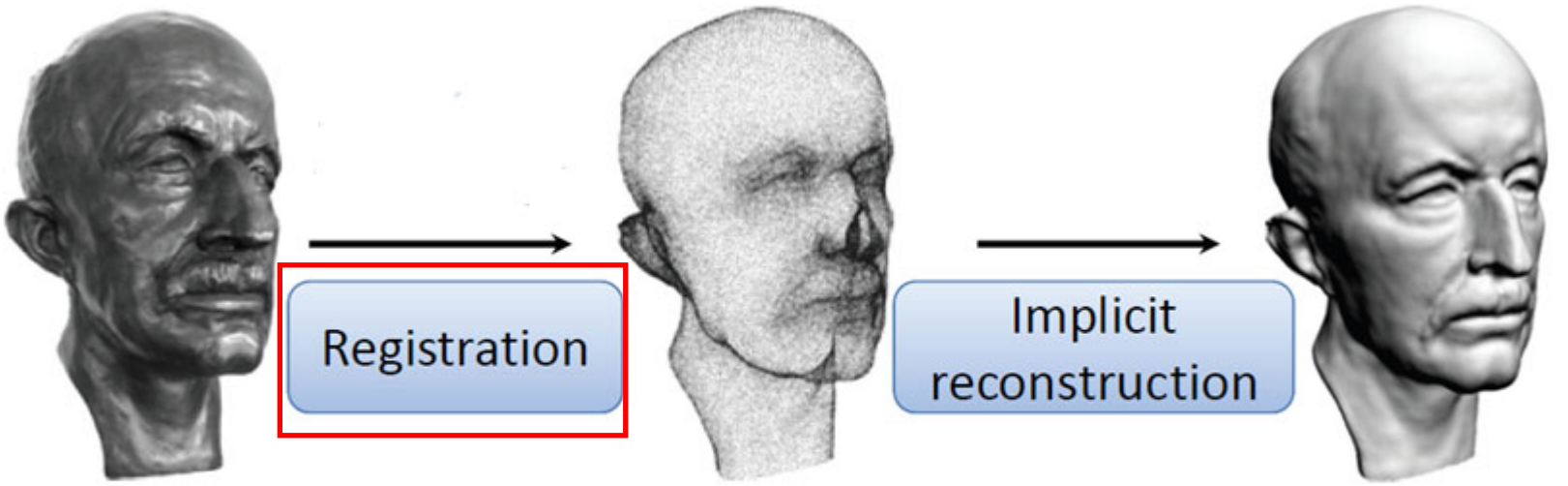
Point clouds have noise, gaps, outliers, and are implausible to display

How to convert them to surfaces?

From Point Clouds to Surfaces

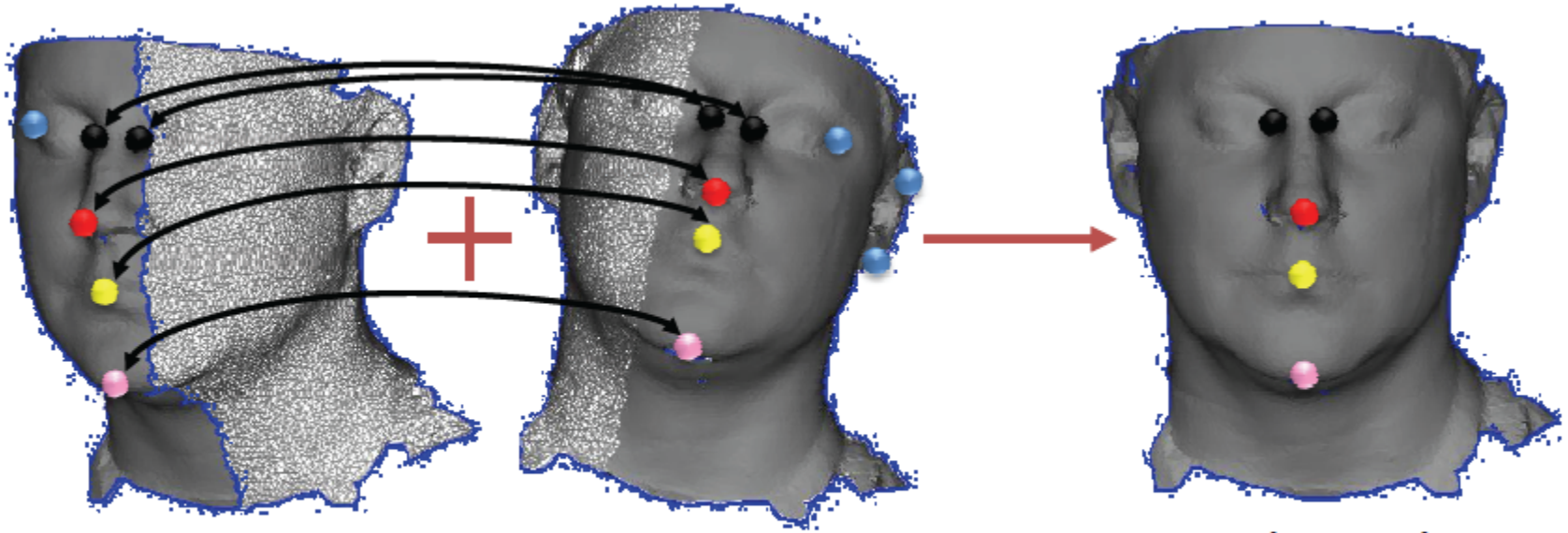


From Point Clouds to Surfaces



Registration

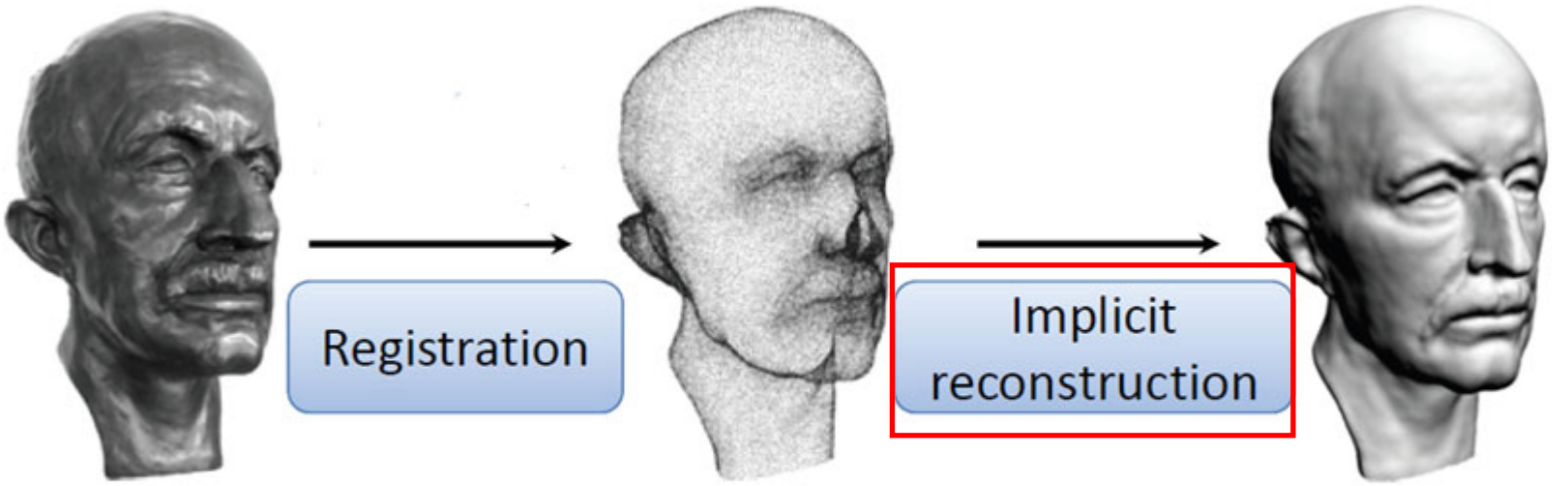
Select some features (landmarks) manually
(or automatically by matching point descriptors)
Then estimate best rotation+translation to align each scan with another



Partially Overlapping Scans

Aligned Scans

From point clouds to meshes



Part II: Shape representations [next time]

- Polygon Meshes
- Parametric Surfaces
- Voxel grids
- Point clouds
- ***Implicit functions***