

Lecture 2: Image Classification with Linear Classifiers

Two basic data-driven approaches to image classification: K-nearest neighbor and linear classifier

challenge:

viewpoint variation, illumination(光照), background clutter(杂乱), occlusion(遮盖), deformation(变形), intraclass variation, context(环境)

Machine Learning 主要步骤

Machine Learning: Data-Driven Approach

1. Collect a dataset of images and labels
2. Use Machine Learning algorithms to train a classifier
3. Evaluate the classifier on new images

Example training set

```
def train(images, labels):
    # Machine learning!
    return model
```

```
def predict(model, test_images):
    # Use model to predict labels
    return test_labels
```

airplane



automobile



bird



cat



deer



Stanford

Nearest Neighbor Classifier

train就是memory the data

predict就是把要预测的点与所有之前train时存下的数据算距离（至于用什么距离下文会谈到），找到最近的点的label就是预测值。如果是k-Nearest Neighbor就是找到k个最近的点，出现最多次的label就是预测值。

所以训练速度：train step O (1); predict O (N)

由此可以看出我们实际不可能使用Nearest Neighbor 算法，我们都是希望开发者花很长的时间把模型训练好，我们使用时可以快速预测才行。

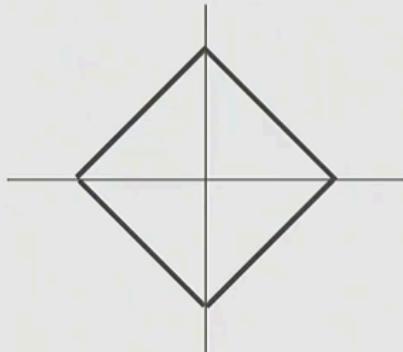
Distance

K-Nearest Neighbors: Distance Metric

PyTorch深度学习课堂

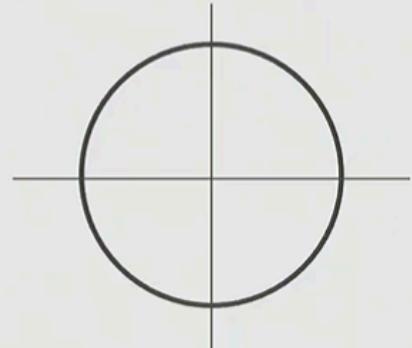
L1 (Manhattan) distance

$$d_1(I_1, I_2) = \sum_p |I_1^p - I_2^p|$$



L2 (Euclidean) distance

$$d_2(I_1, I_2) = \sqrt{\sum_p (I_1^p - I_2^p)^2}$$



怎么选？比如：如果你的feature对旋转敏感，选L1；不敏感选L2

Hyperparameters

Setting Hyperparameters

PyTorch深度学习课堂

Idea #1: Choose hyperparameters that work best on the training data

BAD: K = 1 always works perfectly on training data

train

Idea #2: choose hyperparameters that work best on test data

BAD: No idea how algorithm will perform on new data

train

test

Idea #3: Split data into train, val; choose hyperparameters on val and evaluate on test

Better!

train

validation

test

and then try to

Stanford

要利用validation来选取Hyperparameters。

怎么从test集分出一块作为validation集又是一个问题？于是又提出cross-validation

Setting Hyperparameters

PyTorch深度学习课堂

train

Idea #4: Cross-Validation: Split data into folds, try each fold as validation and average the results

| | | | | |
|--------|--------|--------|--------|--------|
| fold 1 | fold 2 | fold 3 | fold 4 | fold 5 |
| fold 1 | fold 2 | fold 3 | fold 4 | fold 5 |
| fold 1 | fold 2 | fold 3 | fold 4 | fold 5 |
| fold 1 | fold 2 | fold 3 | fold 4 | fold 5 |
| fold 1 | fold 2 | fold 3 | fold 4 | fold 5 |

test

Useful for small datasets, but not used too frequently in deep learning

注意数据量很大时不会这么做，计算成本太高。

Summary

K-Nearest Neighbors: Summary

PyTorch深度学习课堂

In image classification we start with a training set of images and labels, and must predict labels on the test set

The K-Nearest Neighbors classifier predicts labels based on the **K nearest training examples**

Distance metric and K are hyperparameters

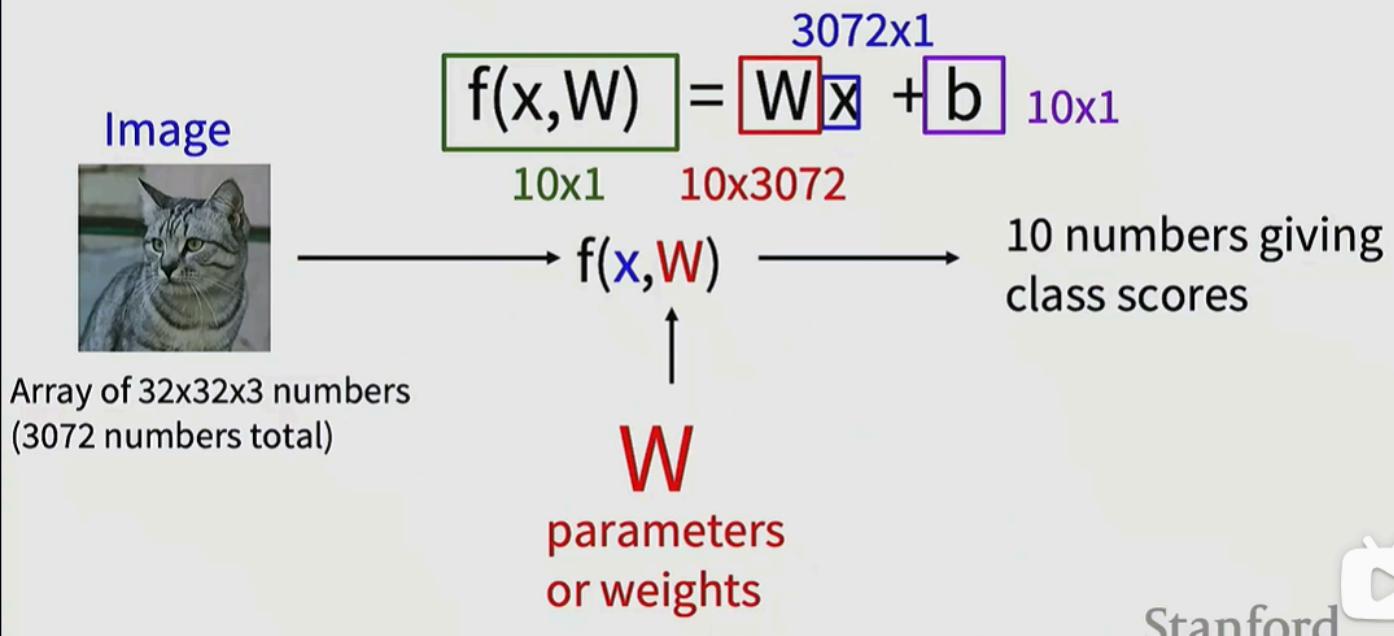
Choose hyperparameters using the validation set

Only run on the test set once at the very end!

Linear Classifier

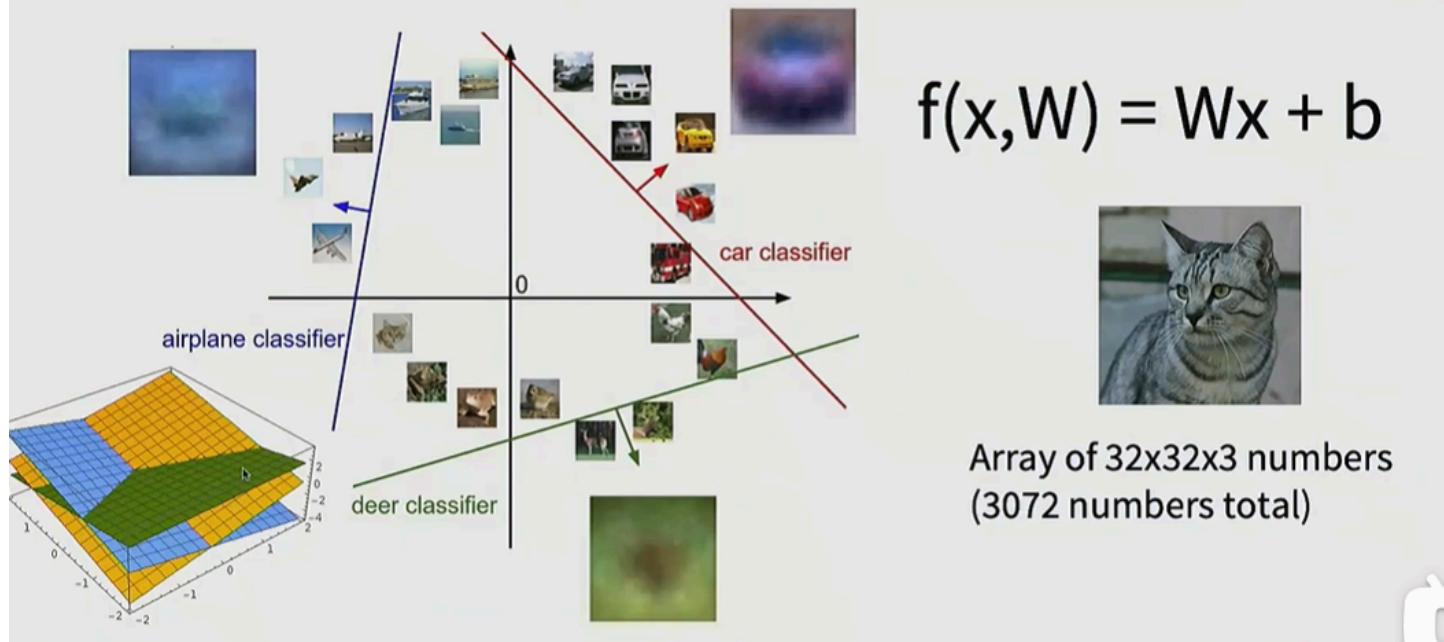
Parametric Approach: Linear Classifier

PyTorch深度学习课堂



几何解释：找到超平面把数据分隔开

Interpreting a Linear Classifier: Geometric Viewpoint



线性分类显然难以解决以下问题：

Hard cases for a linear classifier

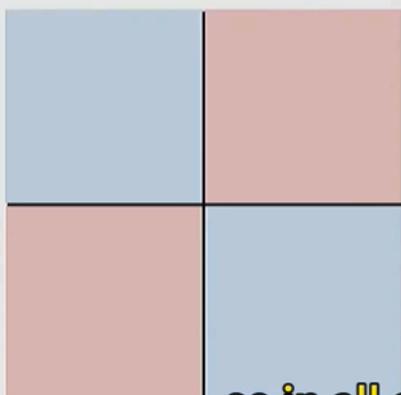
PyTorch深度学习课堂 bilibili

Class 1:

First and third quadrants

Class 2:

Second and fourth quadrants

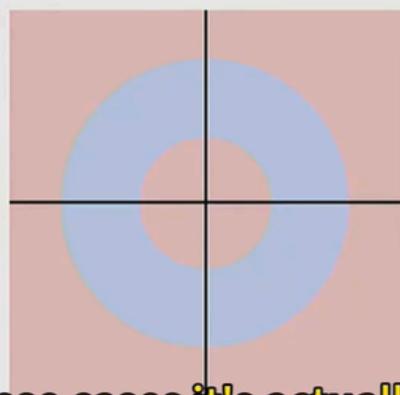


Class 1:

$1 \leq L_2 \text{ norm} \leq 2$

Class 2:

Everything else

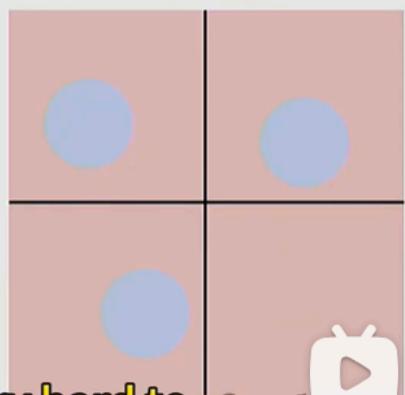


Class 1:

Three modes

Class 2:

Everything else



so in all of these cases it's actually very hard to
所以在所有这些情况下，实际上很难

对于Linear Classifier，本质就是要找到W矩阵使得输入x，能使得结果在正确的分类上分数最大。那W怎么确定？刚开始就是随机初始化了，然后再通过某种方式不断调整。

Linear Classifier – Choose a good W



| | | | |
|------------|------------|-------------|--------------|
| airplane | -3.45 | -0.51 | 3.42 |
| automobile | -8.87 | 6.04 | 4.64 |
| bird | 0.09 | 5.31 | 2.65 |
| cat | 2.9 | -4.22 | 5.1 |
| deer | 4.48 | -4.19 | 2.64 |
| dog | 8.02 | 3.58 | 5.55 |
| frog | 3.78 | 4.49 | -4.34 |
| horse | 1.06 | -4.37 | -1.5 |
| ship | -0.36 | -2.09 | -4.79 |
| truck | -0.72 | -2.93 | 6.14 |

1. Define a loss function that quantifies our unhappiness with the scores across the training data.
2. Come up with a way of efficiently finding the parameters that minimize the loss function. (optimization)

Cat image by Nikita is licensed under CC-BY2.0. Car image is CC0 1.0 public domain. Frog image is in the public domain.

损失函数则用于描述该模型的结果和真实结果之间的差距。

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$



| | | | |
|------|------|-----|------|
| cat | 3.2 | 1.3 | 2.2 |
| car | 5.1 | 4.9 | 2.5 |
| frog | -1.7 | 2.0 | -3.1 |

A loss function tells how good our current classifier is

Given a dataset of examples

$$\{(x_i, y_i)\}_{i=1}^N$$

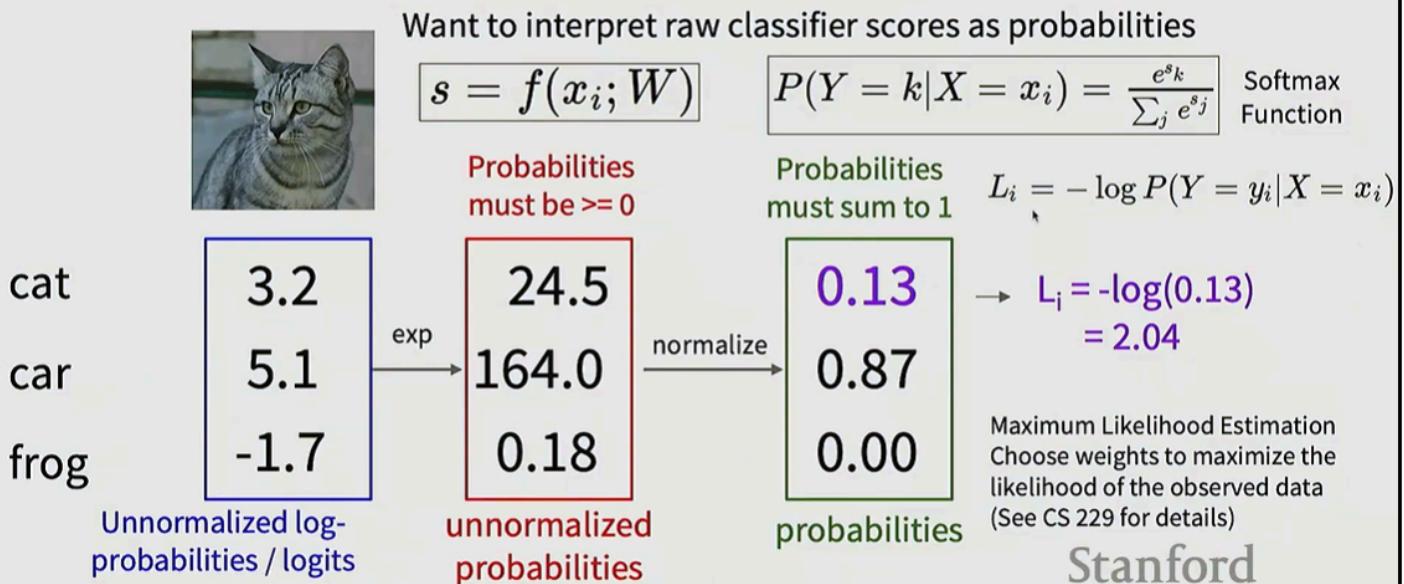
Where x_i is image and y_i is (integer) label

Loss over the dataset is a average of loss over examples:

$$L = \frac{1}{N} \sum_i L_i(f(x_i, W), y_i)$$

显然损失函数有很多种，下面提到的Softmax Classifier（**依然是Linear classifier，虽然过程中有非线性的操作，但本质上决策边界依然是线性的**），则用了 $-\log(\text{softmax function})$, softmax可以把 $Wx+b$ 算出来的分数转化为0~1看成概率，并且最理想的情况就应该是一张图片被分类到正确的类概率为1，其他类全为0，而 $-\log(1)$ 刚好为0，所以我们只需要最小化损失函数 L 即可

Softmax Classifier (Multinomial Logistic Regression)



Stanford CS231n

这里 L_i 最小到0，最大到正无穷。注意 $L_i = -\log P(Y = y_i | X = x_i)$, 只计算了正确分类所对应的值！

Softmax Classifier (Multinomial Logistic Regression)

Want to interpret raw classifier scores as probabilities



$$s = f(x_i; W)$$

$$P(Y = k|X = x_i) = \frac{e^{s_k}}{\sum_j e^{s_j}}$$

Softmax Function

Maximize probability of correct class

$$L_i = -\log P(Y = y_i|X = x_i)$$

Putting it all together:

$$L_i = -\log\left(\frac{e^{s_{y_i}}}{\sum_j e^{s_j}}\right)$$

| | |
|------|------|
| cat | 3.2 |
| car | 5.1 |
| frog | -1.7 |

Q2: At initialization all s will be approximately equal; what is the loss?
A: $-\log(1/C) = \log(C)$,
If $C = 10$, then $L_i = \log(10) \approx 2.3$



Stanford

这里给出了一个验证自己代码是否正确的方式，如果刚开始初始化时把参数基本设为基本相同的，那 L_i 应该就在2.3左右

另外，单个样本的交叉熵就等于 L_i

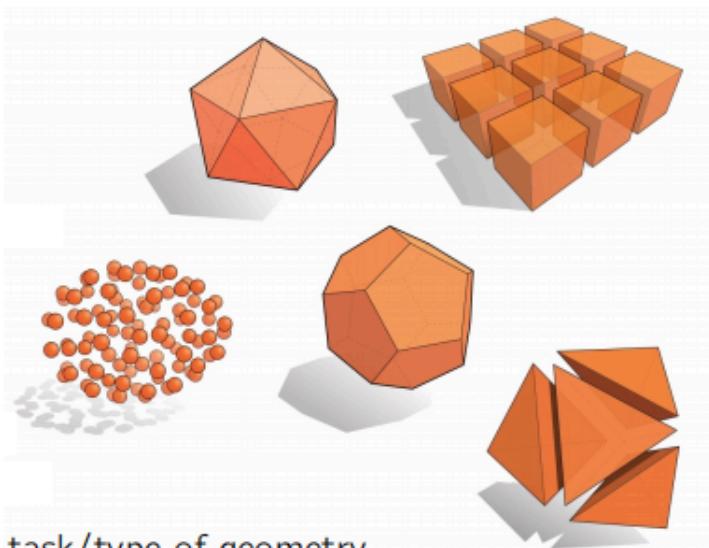
并且**交叉熵=熵+KL散度**，而对于分类问题，正确分布是one-hot型的，熵很低，基本为0，所以可以说认为任务就是最小化KL散度

Lecture 3D version

how to represent 3D?

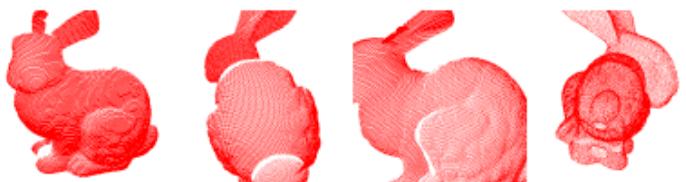
Many Ways to Represent Geometry

- Explicit
 - Point cloud
 - Polygon mesh
 - Subdivision, NURBS
 - ...
- Implicit
 - Level sets
 - Algebraic surface
 - Distance functions
 - ...
- Each choice best suited to a different task/type of geometry

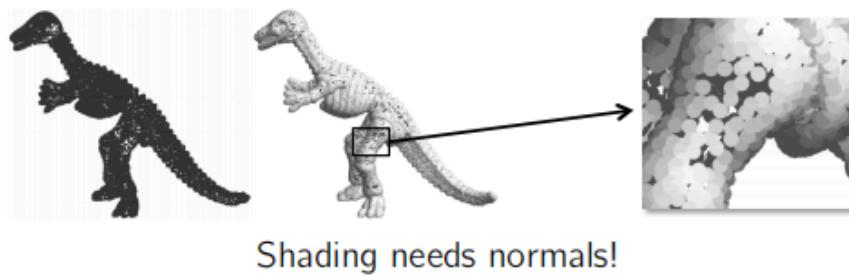


Slide credit: Ren Ng

Point Clouds



- Simplest representation: **only points**, no connectivity
- Collection of (x, y, z) coordinates, possibly with normal
- Points with orientation are called **surfels**



point clouds advantages and disadvantages

Point Clouds

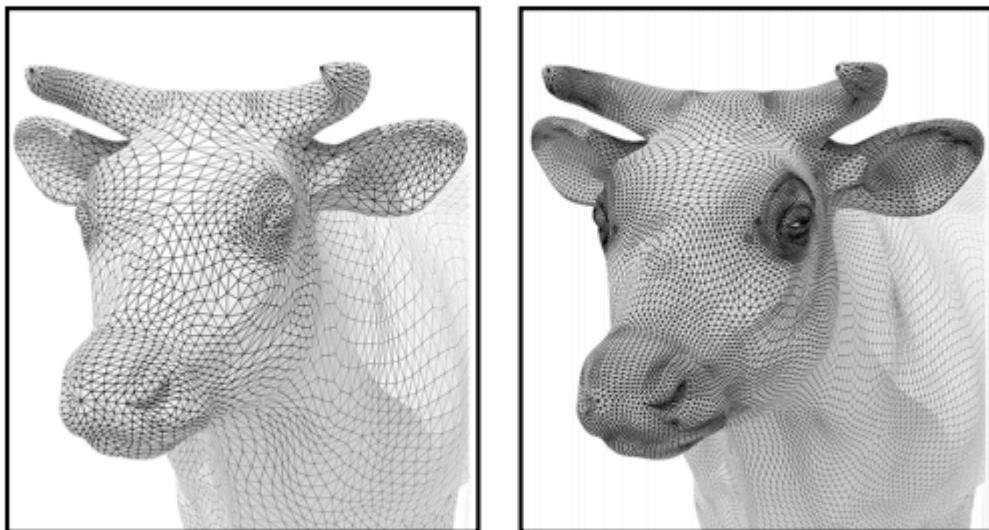


- Easily represent any kind of geometry
- Useful for large datasets
- Difficult to draw in undersampled regions
- Other limitations:
 - No simplification or subdivision
 - No direction smooth rendering
 - No topological information



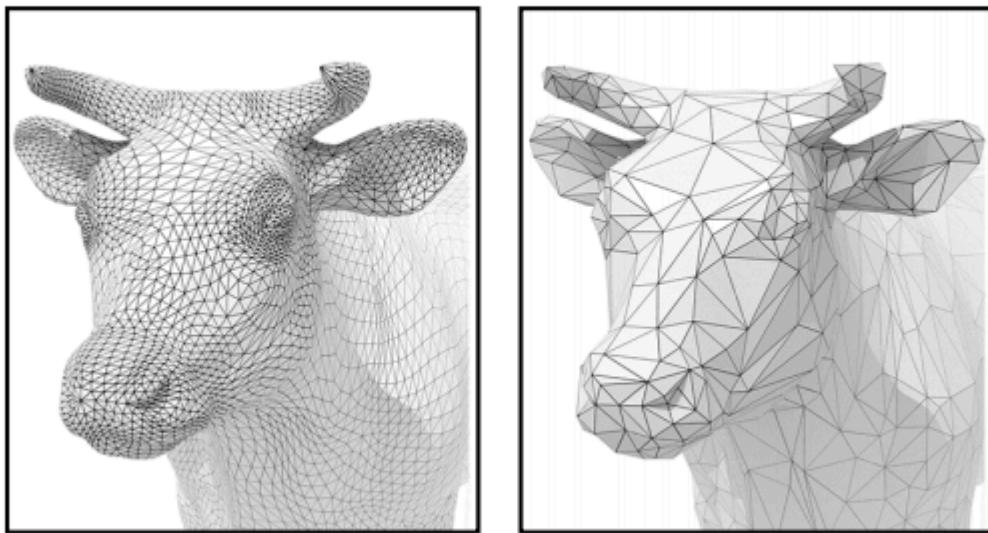
Polygonal Meshes

Mesh Upsampling - Subdivision



Increase resolution via interpolation

Mesh Downsampling - Simplification



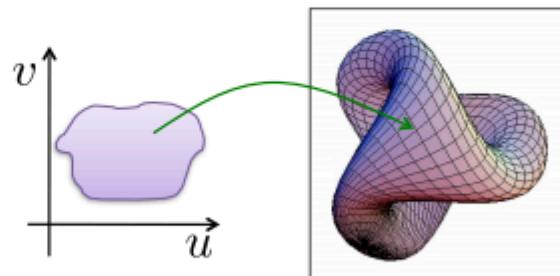
Decrease resolution; try to preserve shape/appearance

Parametric Representation

Parametric Representation

Range of a function $f : X \rightarrow Y, X \subseteq \mathbb{R}^m, Y \subseteq \mathbb{R}^n$

Surface in 3D: $m = 2, n = 3$

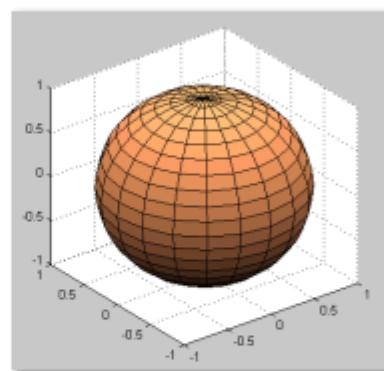


$$s(u, v) = (x(u, v), y(u, v), z(u, v))$$

Parametric Surfaces

Sphere in 3D

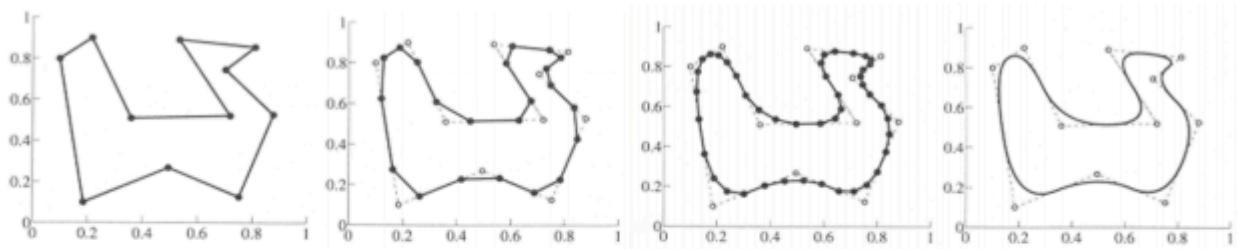
$$s : \mathbb{R}^2 \rightarrow \mathbb{R}^3$$



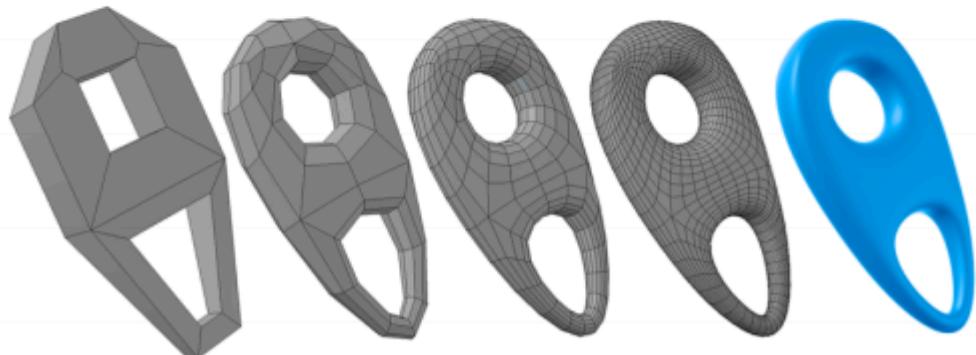
$$s(u, v) = r (\cos(u) \cos(v), \sin(u) \cos(v), \sin(v))$$

$$(u, v) \in [0, 2\pi) \times [-\pi/2, \pi/2]$$

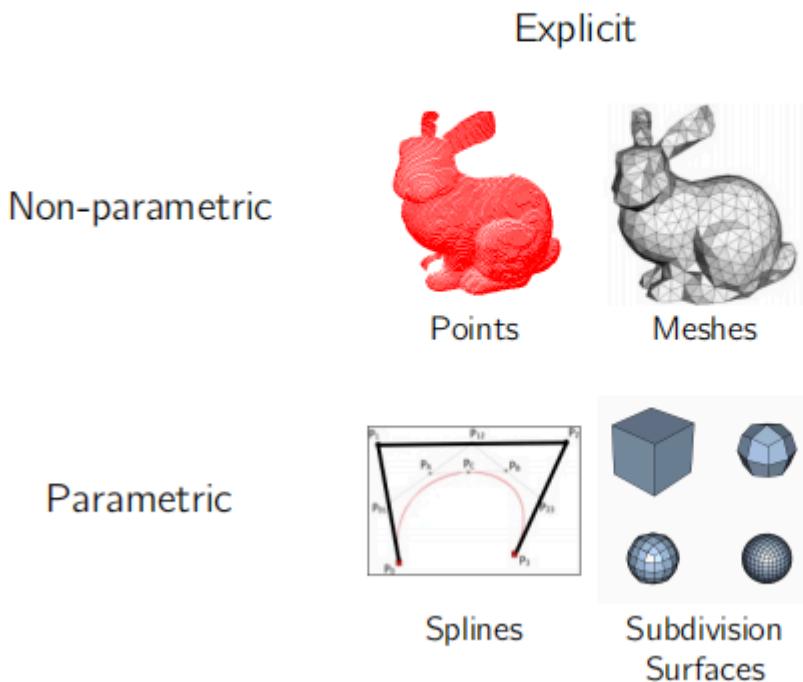
Subdivision Curves/Surfaces



Slide cribbed from Keenan Crane, cribbed from Don Fussell.



Shape Representations

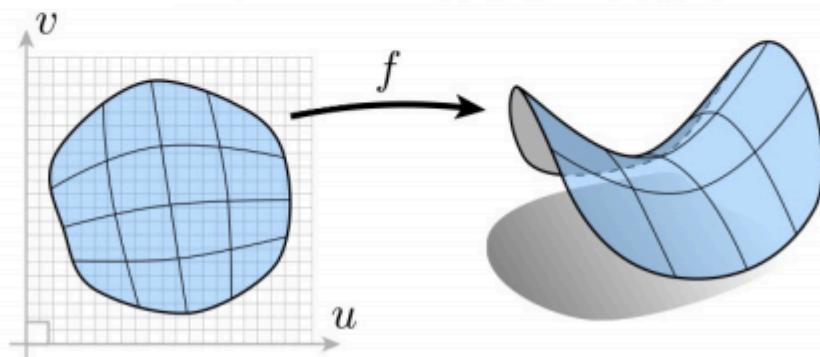


对于显示参数化表达式，是容易采样的，随便给定几个自变量，带入方程计算函数值即可。但任意给定一个点，很难判断这个点在图形内部还是外部（知道点在内外部对于渲染的时候是重要的！）。隐式参数化则刚好相反。

“Explicit” Representations of Geometry

All points are given directly.

Generally: $f : \mathbb{R}^2 \rightarrow \mathbb{R}^3; (u, v) \mapsto (x, y, z)$

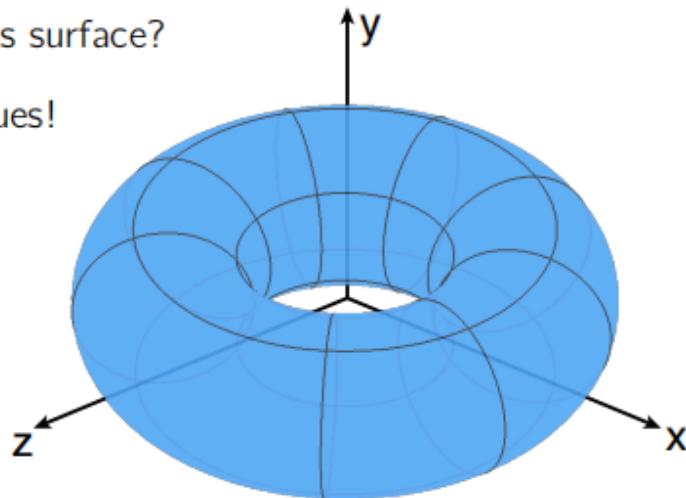


Explicit Surface – Sampling Is Easy

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

What points lie on this surface?

Just plug in (u, v) values!



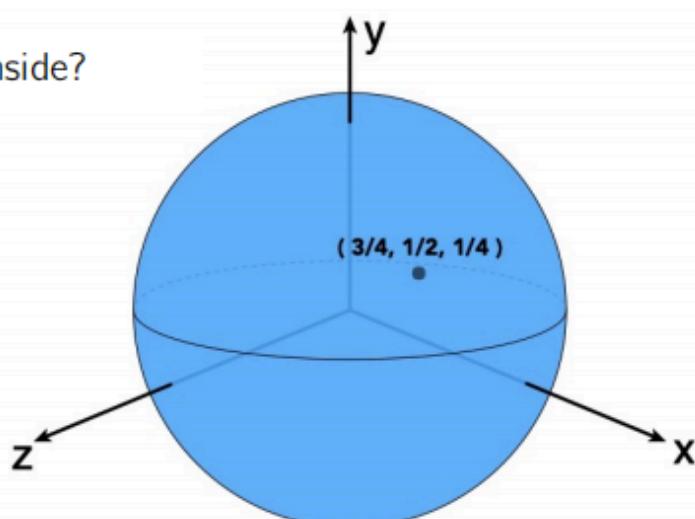
Explicit representations make some tasks easy.

Slide credit: Re

Explicit Surface – Inside/Outside Test Hard

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

Is $(3/4, 1/2, 1/4)$ inside?



Some tasks are hard with explicit representations.

Slide credit: Re

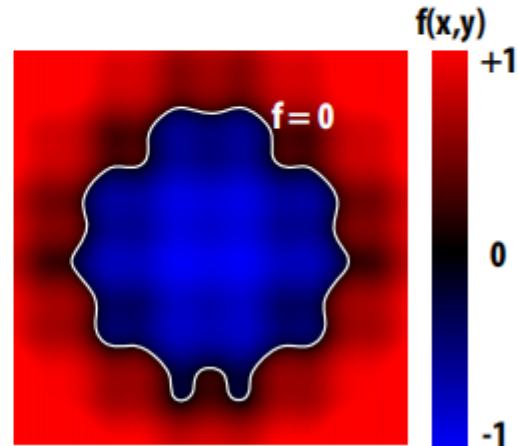
“Implicit” Representations of Geometry

Based on classifying points

- Points satisfy some specified relationship.

E.g., sphere: all points in 3D, where $x^2 + y^2 + z^2 = 1$

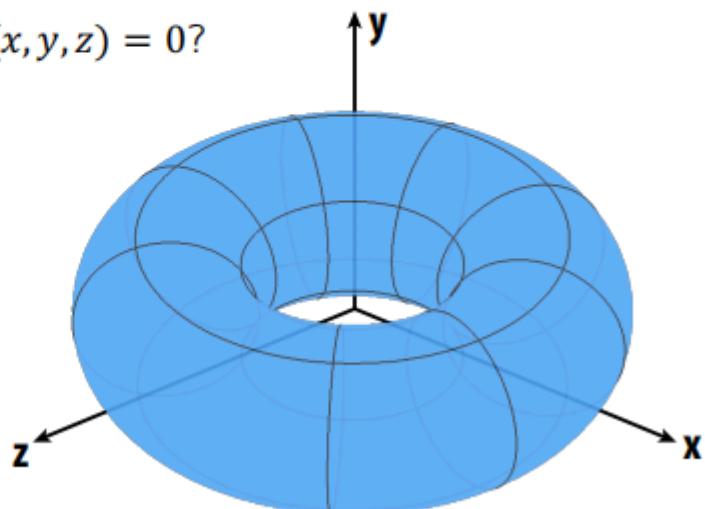
More generally, $f(x, y, z) = 0$



Implicit Surface – Sampling Can Be Hard

$$f(x, y, z) = (2 - \sqrt{x^2 + y^2})^2 + z^2 - 1$$

What points lie on $f(x, y, z) = 0$?



Some tasks are hard with implicit representations.

...

Implicit Surface – Inside/Outside Tests Easy

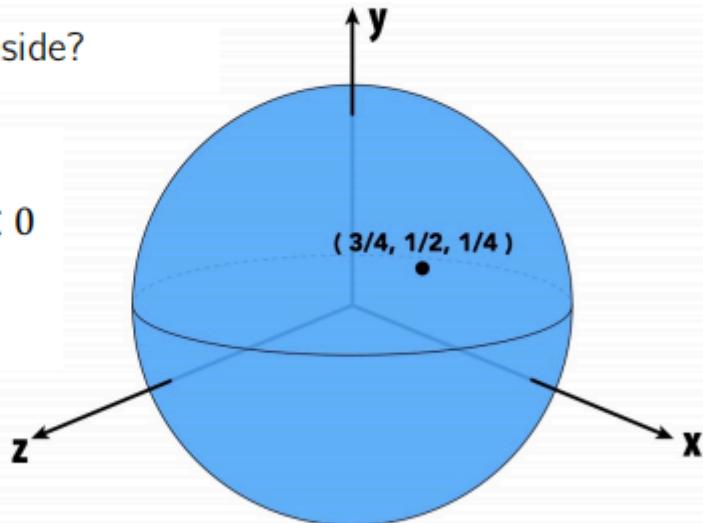
$$f(x, y, z) = x^2 + y^2 + z^2 - 1$$

Is $(3/4, 1/2, 1/4)$ inside?

Just plug it in:

$$f(x, y, z) = -1/8 < 0$$

Yes, inside.



Implicit representations make some tasks easy.

Slide credit:

对于复杂的图像，需要用到布尔运算

Algebraic Surfaces (Implicit)

Surface is zero set of a polynomial in x, y, z .



$$x^2 + y^2 + z^2 = 1$$



$$(R - \sqrt{x^2 + y^2})^2 + z^2 = r^2$$



$$(x^2 + \frac{9y^2}{4} + z^2 - 1)^3 =$$

$$x^2 z^3 + \frac{9y^2 z^3}{80}$$

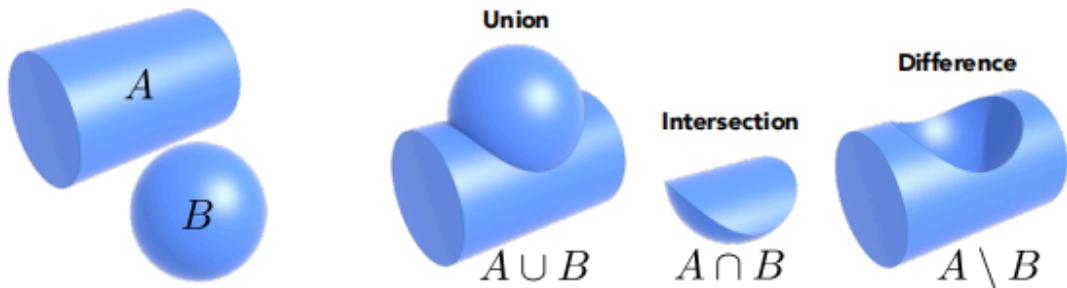


More complex shapes?

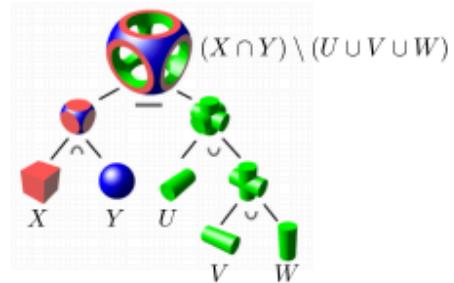
Slide credit:

Constructive Solid Geometry (Implicit)

Combine implicit geometry via Boolean operations



Boolean expressions:



Slide credit: Re

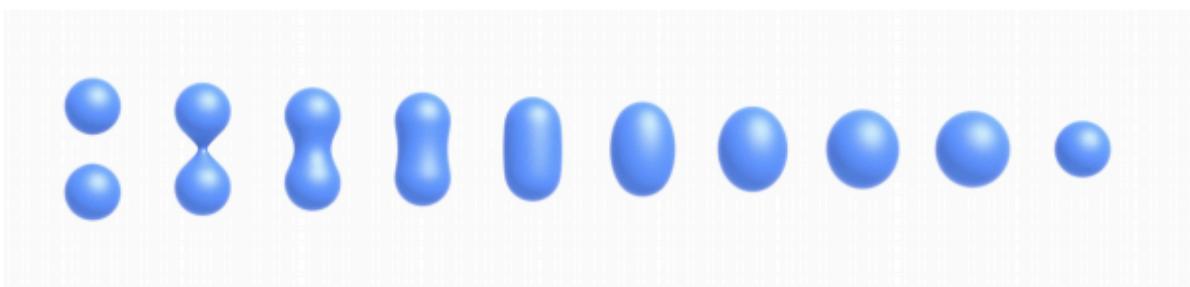
其他隐式方法：

Distance Functions (Implicit)

Instead of Boolean, gradually blend surfaces together using

Distance functions:

Giving minimum distance (could be **signed** distance) from anywhere to object



Scene of Pure Distance Functions (Not Easy!)



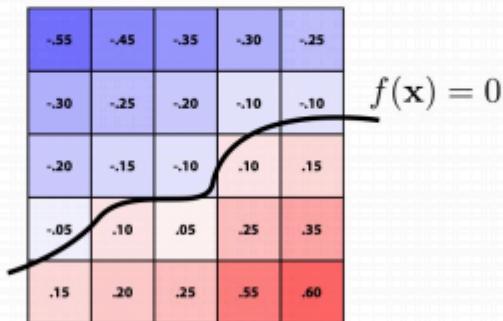
See <http://iquilezles.org/www/material/nvscene2008/nvscene2008.htm>

Level Set Methods (Implicit)

Implicit surfaces have some nice features (e.g., merging/splitting).

But hard to describe complex shapes in closed form

Alternative: store a grid of values approximating function

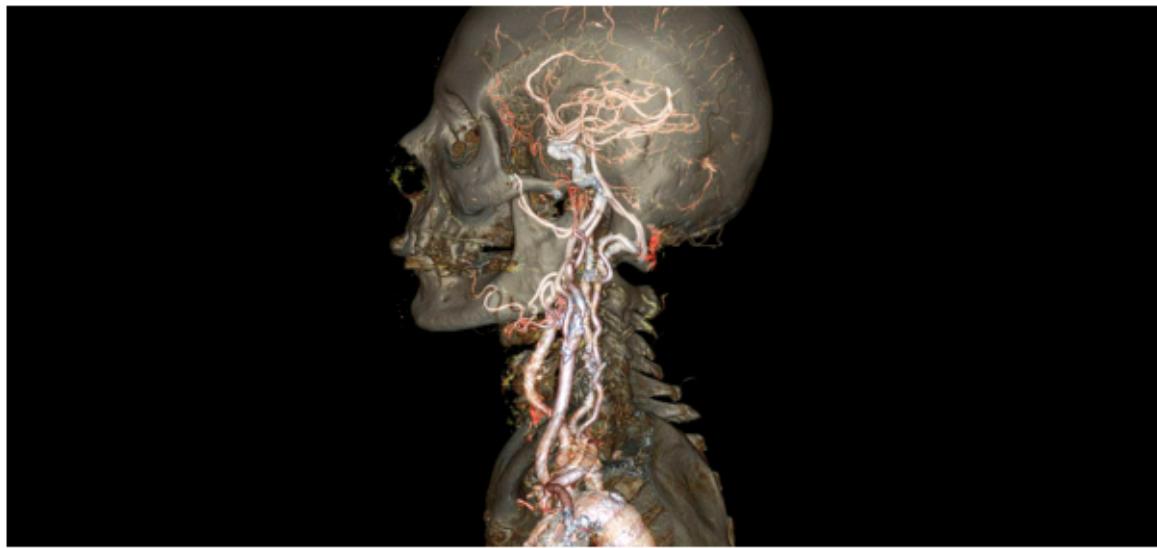


Surface is found where interpolated values equal zero.

Provides much more explicit control over shape (like a texture)

Slide credit:

Level Sets from Medical Data (CT, MRI, etc.)

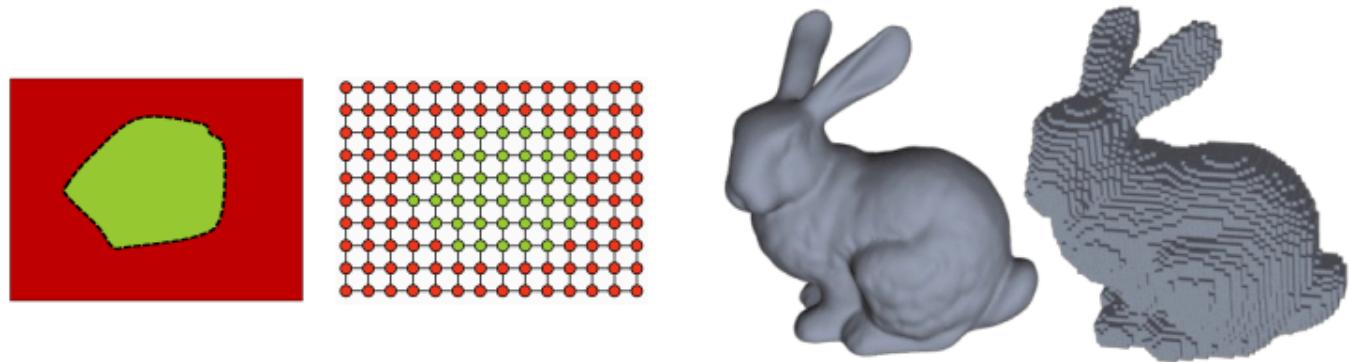


Level sets encode, e.g., constant tissue density

Slide credit: Dan L.

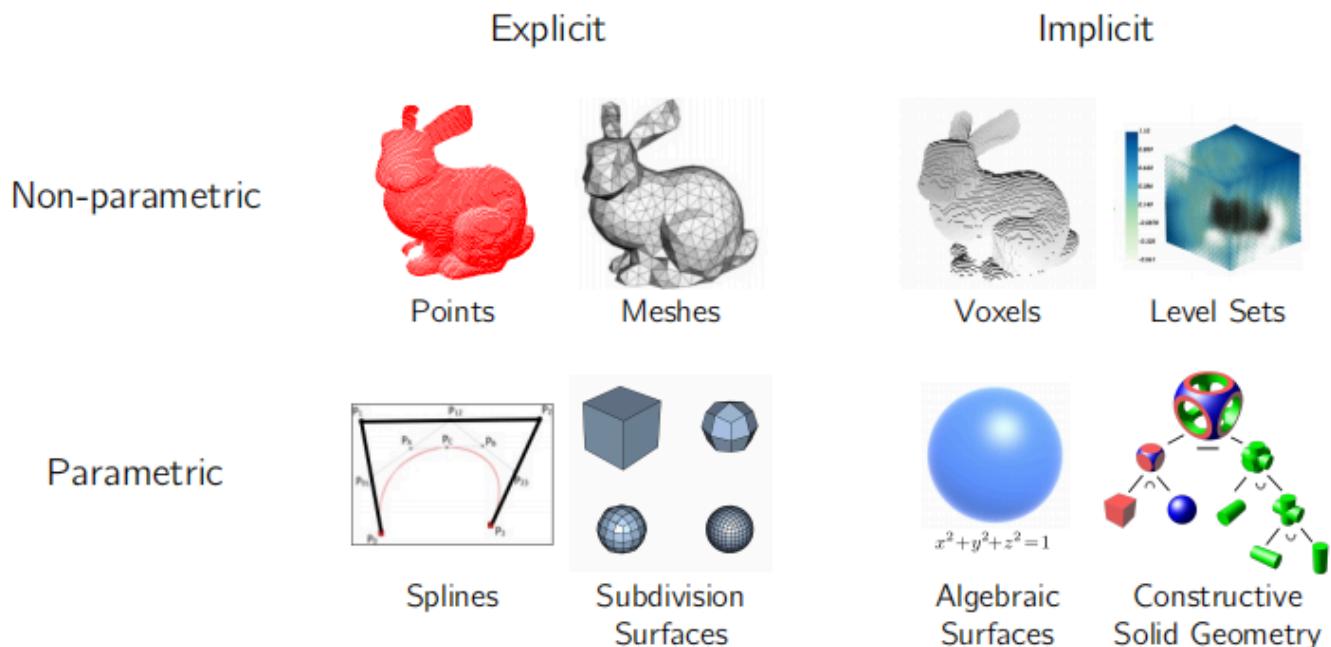
Related Representation: Voxels

- Binary thresholding the volumetric grid



总结：

Shape Representations



dataset: ShapeNet CO3D

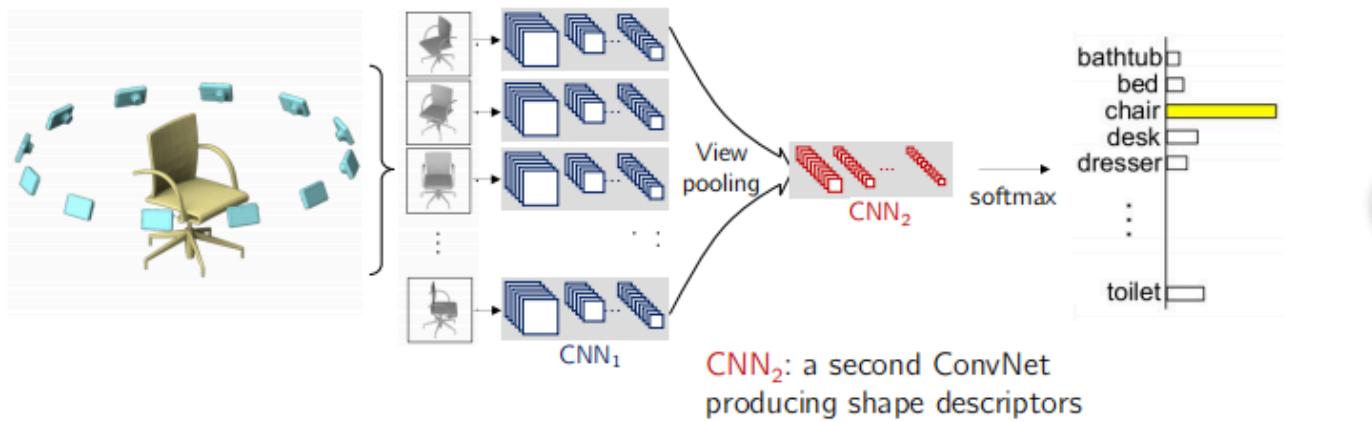
目前的tasks:

AI + Geometry: Tasks

- $P(S)$ or $P(S|c)$ --- Generative models
 - Learning (conditional) shape priors
 - Shape generation, completion, & geometry data processing
- $P(c|S)$ --- Discriminative models
 - Learning shape descriptors
 - Shape classification, segmentation, view estimation, etc.
- Joint modeling of 3D and 2D data
 - Large-scale 2D datasets & very good pretrained models
 - Differentiable projection/back-projection & differentiable/neural rendering
- Joint modeling of multi-modal data beyond visual (e.g., text)

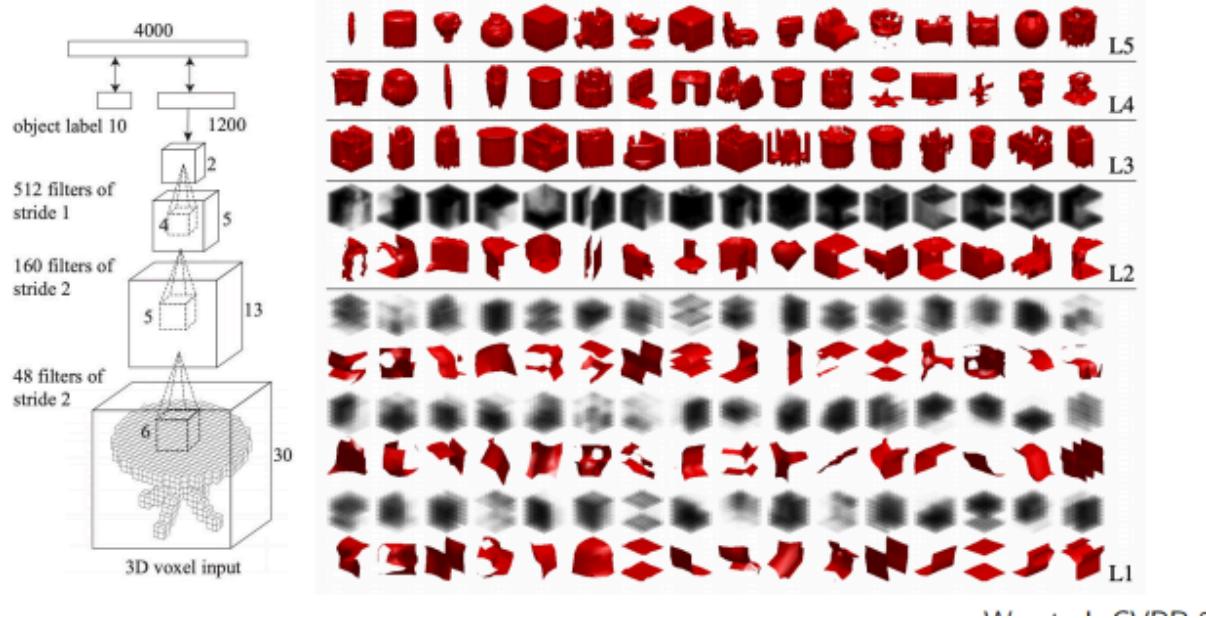
由于2d情况的已经很成熟，所以我们可以借鉴或者利用2d的情况

Multi-View CNN



Pixels -> Voxels

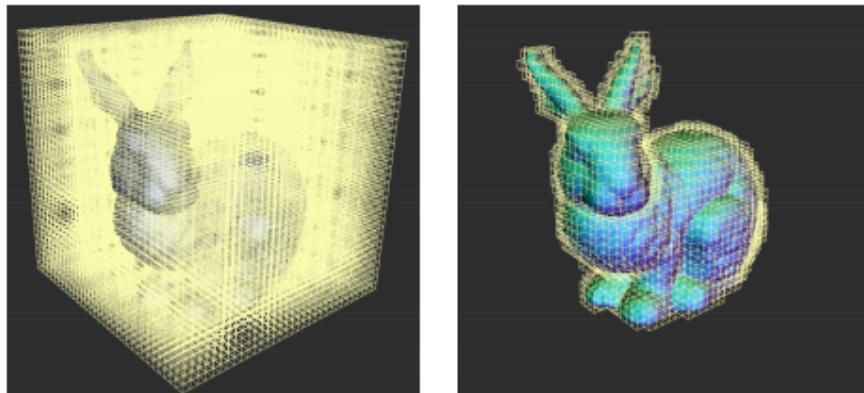
- 3D Conv Deep Belief Networks (CDBN)



其他方法：

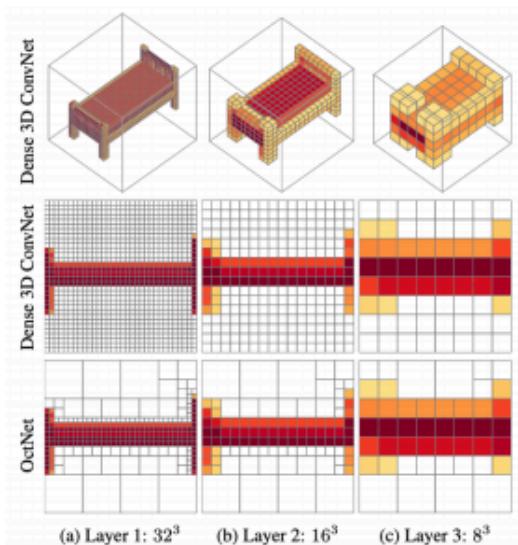
Octave Tree Representations

- Store the sparse surface signals
- Constrain the computation near the surface

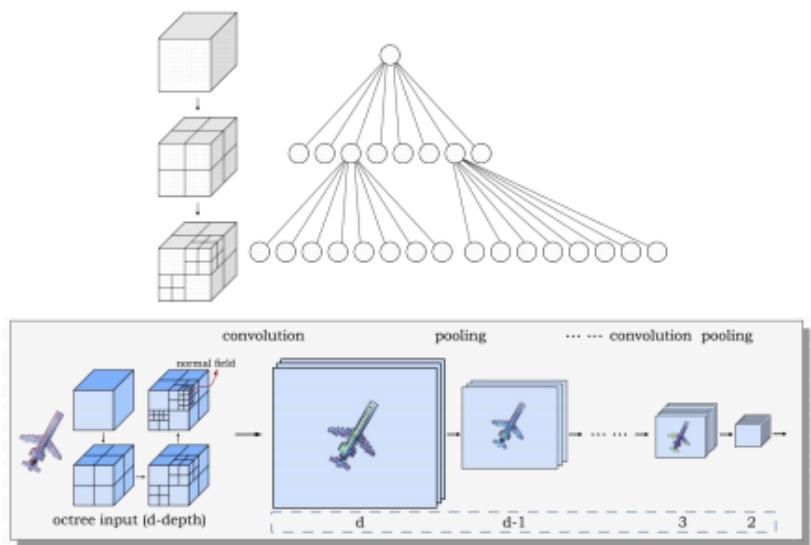


Slide Credit: Hao Su

Octree: Recursively Partition the Space



Riegler et al. OctNet. CVPR 2017



Wang et al. O-CNN. SIGGRAPH 2017

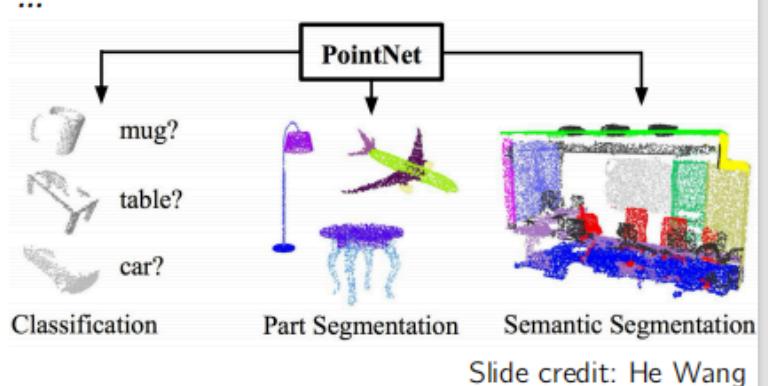
PointNet: Learning on Point Clouds



End-to-end learning for irregular point data

Unified framework for various tasks

Charles R. Qi, Hao Su, Kaichun Mo, Leonidas J. Guibas.
PointNet: Deep Learning on Point Sets for 3D
Classification and Segmentation. (CVPR'17)



考虑物体的对称型，并且point cloud的特性

Invariances

The model has to respect key desiderata for point clouds:

Point Permutation Invariance

Point cloud is a set of **unordered** points

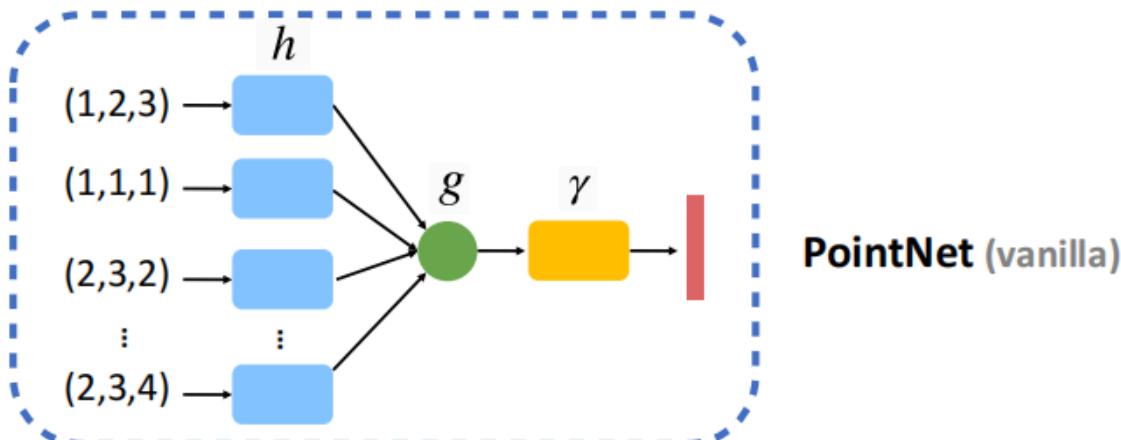
Sampling Invariance

Output a function of the underlying geometry and **not the sampling**

所以我们也考虑

Construct Symmetric Functions by NNs

$f(x_1, x_2, \dots, x_n) = \gamma \circ g(h(x_1), \dots, h(x_n))$ is symmetric if g is symmetric



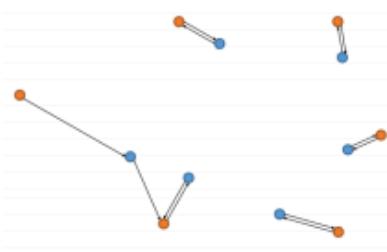
Slide credit: He Wang

由于此时output是point cloud不再是分类的标签or体积块，不能用交叉熵做loss function 所以我们需要引入新的方式计算output的point cloud和真值的差异

Distance Metrics for Point Clouds

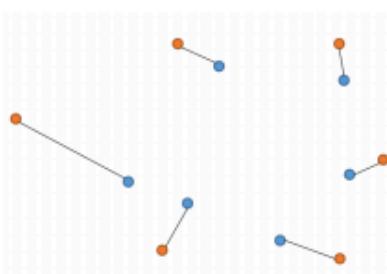
Chamfer distance We define the Chamfer distance between $S_1, S_2 \subseteq \mathbb{R}^3$ as:

$$d_{CD}(S_1, S_2) = \sum_{x \in S_1} \min_{y \in S_2} \|x - y\|_2 + \sum_{y \in S_2} \min_{x \in S_1} \|x - y\|_2$$



Earth Mover's distance Consider $S_1, S_2 \subseteq \mathbb{R}^3$ of equal size $s = |S_1| = |S_2|$. The EMD between A and B is defined as:

$$d_{EMD}(S_1, S_2) = \min_{\phi: S_1 \rightarrow S_2} \sum_{x \in S_1} \|x - \phi(x)\|_2$$



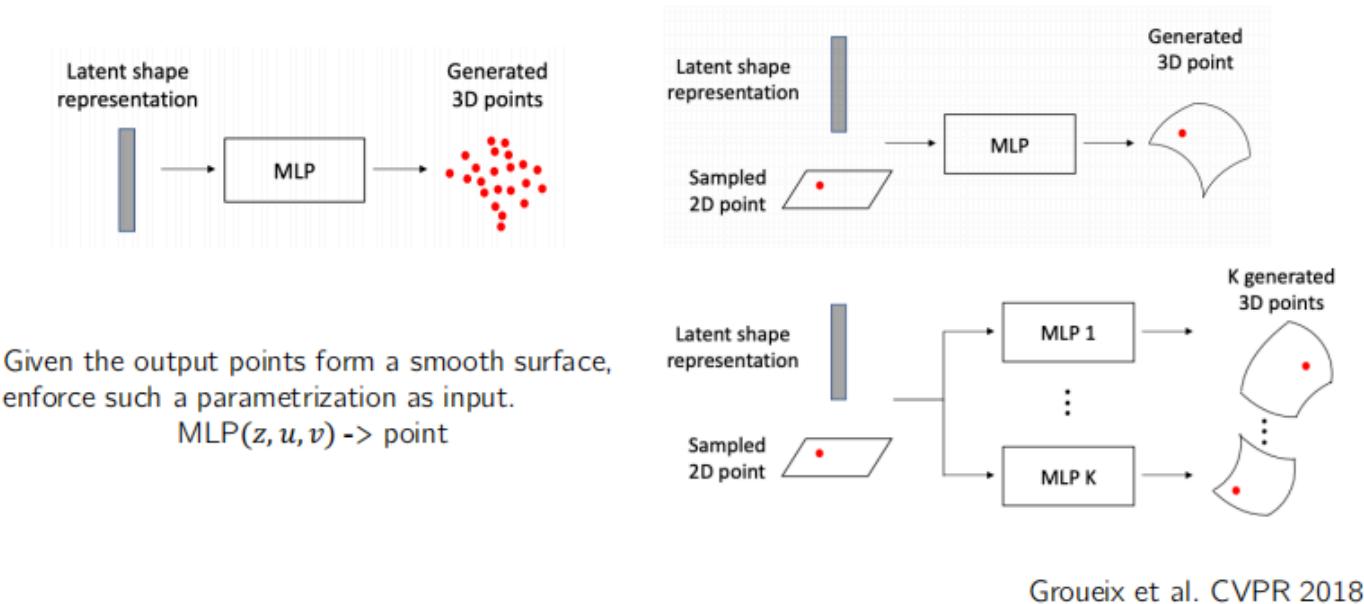
where $\phi : S_1 \rightarrow S_2$ is a bijection.

A Point Set Generation Network for 3D Object Reconstruction from a Single Image, CVPR 2016

Slide credit: He Wang

AltasNet想获得更光滑的曲面

Parametric Decoder: AtlasNet



后续还涉及到了Deep Implicit Functions, Reconstruction & Novel View Synthesis with NeRF等