3D Perception: PointNet and NERFs

Sanjiban Choudhury
Last Class: How does a robot identify objects?
Last Class: How does a robot identify objects?

Classification

CAT

No spatial extent

Semantic Segmentation

GRASS, CAT, TREE, SKY

No objects, just pixels

Object Detection

DOG, DOG, CAT

Multiple Object

Instance Segmentation

DOG, DOG, CAT

Slides from Stanford CS231N: Object Detection and Image Segmentation
But manipulating objects require 3D reasoning!
Depth cameras give us 3D information!
Masked Depth Image -> Point Cloud
Activity!
Think-Pair-Share!

Think (30 sec): Given a point cloud of an object, how would you learn where to grasp it? What are some informative features?

Pair: Find a partner

Share (45 sec): Partners exchange ideas
Can we learn effective feature learning directly on point clouds?
PointNet

End-to-end learning for **scattered, unordered** point data

PointNet

End-to-end learning for **scattered, unordered** point data

Unified framework for various tasks

Object Classification
Object Part Segmentation
Semantic Scene Parsing...

PointNet

End-to-end learning for **scattered, unordered** point data

**Unified** framework for various tasks

---

Classification | Part Segmentation | Semantic Segmentation

---

Slides from Qi et al, CVP 2017  
http://stanford.edu/~rqi/pointnet/docs/cvpr17_pointnet_slides.pdf
Why is learning with point clouds challenging?
Two Challenges

Challenge 1: **Unordered** point set as input

*Model needs to be invariant to $N!$ permutation*

Challenge 2: Invariance under **geometric** transformations

*Point cloud rotation should not alter classification results*
Two Challenges

Challenge 1: **Unordered** point set as input

*Model needs to be invariant to $N!$ permutation*

Challenge 2: Invariance under **geometric** transformations

*Point cloud rotation should not alter classification results*
Unordered Input

Point cloud: N orderless points, each represented by a D dim vector

Slides from Qi et al, CVP 2017  http://stanford.edu/~rgi pointnet/docs/cvpr17__pointnet_slides.pdf
Unordered Input

Point cloud: $N$ orderless points, each represented by a $D$ dim vector

represents the same set as

Point cloud: $N$ orderless points, each represented by a $D$-dim vector

Model needs to be invariant to $N!$ permutations

Permutation Invariance: Symmetric Function

\[ f(x_1, x_2, \ldots, x_n) \equiv f(x_{\pi_1}, x_{\pi_2}, \ldots, x_{\pi_n}), \quad x_i \in \mathbb{R}^D \]
Permutation Invariance: Symmetric Function

\[ f(x_1, x_2, \ldots, x_n) \equiv f(x_{\pi_1}, x_{\pi_2}, \ldots, x_{\pi_n}), \quad x_i \in \mathbb{R}^D \]

Examples:

\[ f(x_1, x_2, \ldots, x_n) = \max\{x_1, x_2, \ldots, x_n\} \]

\[ f(x_1, x_2, \ldots, x_n) = x_1 + x_2 + \ldots + x_n \]

\[ \text{...} \]

Permutation Invariance: Symmetric Function

\[ f(x_1, x_2, \ldots, x_n) \equiv f(x_{\pi_1}, x_{\pi_2}, \ldots, x_{\pi_n}), \quad x_i \in \mathbb{R}^D \]

Examples:

\[ f(x_1, x_2, \ldots, x_n) = \max \{x_1, x_2, \ldots, x_n\} \]
\[ f(x_1, x_2, \ldots, x_n) = x_1 + x_2 + \ldots + x_n \]

\[ \ldots \]

How can we construct a family of symmetric functions by neural networks?

Observe:

\[ f(x_1, x_2, \ldots, x_n) = \gamma \circ g(h(x_1), \ldots, h(x_n)) \] is symmetric if \( g \) is symmetric.
Permutation Invariance: Symmetric Function

Observe:

\[ f(x_1, x_2, \ldots, x_n) = \gamma \circ g(h(x_1), \ldots, h(x_n)) \] is symmetric if \( g \) is symmetric

Slides from Qi et al, CVP 2017  
http://stanford.edu/~rqi/pointnet/docs/cvpr17_pointnet_slides.pdf
Permutation Invariance: Symmetric Function

Observe:

\[ f(x_1, x_2, \ldots, x_n) = \gamma \circ g(h(x_1), \ldots, h(x_n)) \]
is symmetric if \( g \) is symmetric

Permutation Invariance: Symmetric Function

Observe:

\[ f(x_1, x_2, \ldots, x_n) = \gamma \circ g(h(x_1), \ldots, h(x_n)) \] is symmetric if \( g \) is symmetric

### Notes

- **Permutation Invariance**: Symmetric function \( f \) is invariant under permutations of its arguments. It can be expressed as a composition of \( g \) and \( h \), where \( g \) is a simple symmetric function and \( h \) maps each input set \( x_i \) to a single output set.

### Diagram

- **PointNet (vanilla)**: This diagram visualizes the mapping of input sets \((1,2,3), (1,1,1), (2,3,2), \ldots, (2,3,4)\) through the functions \( h, g, \) and \( \gamma \) to output sets, illustrating the permutation invariance.

### References

- Slides from Qi et al, CVP 2017
- [http://stanford.edu/~rqi/pointnet/docs/cvpr17_pointnet_slides.pdf](http://stanford.edu/~rqi/pointnet/docs/cvpr17_pointnet_slides.pdf)
Empirically, we use **multi-layer perceptron (MLP)** and **max pooling**: 

![Diagram of PointNet architecture](http://stanford.edu/~rqi/pointnet/docs/cvpr17_pointnet_slides.pdf)
Two Challenges

Challenge 1: **Unordered** point set as input

*Model needs to be invariant to $N!$ permutation*

Challenge 2: Invariance under **geometric** transformations

*Point cloud rotation should not alter classification results*
Idea: Data dependent transformation for automatic alignment

The transformation is just matrix multiplication!

PointNet Classification Network

PointNet Classification Network

PointNet Classification Network

PointNet Classification Network

input points

nx3

input transform

nx3

T-Net
3x3 transform

matrix multiply

mlp (64,64)

shared

feature transform

nx64

T-Net
64x64 transform

matrix multiply

mlp (64,128,1024)

shared

nx1024

max pool

1024
global feature

mlp
(512,256,k)

output scores

Results on Object Part Segmentation

Slides from Qi et al, CVP 2017  
http://stanford.edu/~rqi/pointnet/docs/cvpr17_pointnet_slides.pdf
Results on Semantic Scene Parsing

Input

Output

How do we use this for learning grasping?
PointNetGPD: Detecting Grasp Configurations from Point Sets

Grasp Dataset

Robot Initial State  Grasp Candidates Generation

Quality Evaluation with PointNet  Best Grasp

Executed Grasp

Liang et al.
PointNetGPD: Detecting Grasp Configurations from Point Sets

Liang et al.
But ... what if we don’t have RGBD data?
Reasons for not having depth data

Don’t have a depth sensor!

Outside in the sun / beyond maximum range

Glass or transparent objects
Glass or transparent objects

Real-world Scene

RealSense D410 Depth Image
Glass or transparent objects

Dishwasher Real-world Scene

RealSense D410 Depth Image
Let’s say I just have a set of images & camera poses

$x_1, y_1, z_1, \theta_1, \phi_1$

$x_2, y_2, z_2, \theta_2, \phi_2$

$x_3, y_3, z_3, \theta_3, \phi_3$
How do we predict a 3D structure?

2D images

Camera Poses

3D structure

$x_1, y_1, z_1, \theta_1, \phi_1$

$x_2, y_2, z_2, \theta_2, \phi_2$

$x_3, y_3, z_3, \theta_3, \phi_3$
If you can predict how the object will look from a novel viewpoint, you have *implicitly* modeled the 3D structure.
Let’s setup a learning problem

\[ x, y, z, \theta, \phi \]

Novel viewpoint

Neural Network

Predicted
Let’s setup a learning problem

\[ x, y, z, \theta, \phi \]

Novel viewpoint

Neural Network

Predicted

Ground Truth
Simple idea:

Can’t we just make the neural network predict a 3D voxel grid of RGB values?
Predict a 3D grid, render image from viewpoint

$x, y, z, \theta, \phi$

Novel viewpoint

Neural Network

Predicted
Predict a 3D grid, render image from viewpoint

$x, y, z, \theta, \phi$

Novel viewpoint

Neural Network

Predicted

What are challenges with this approach?
Challenges

Discretization loses information!

Memory Inefficient!

Not differentiable! (Not a continuous projection)
NERF to the rescue!
What are Neural Radiance Fields (NeRFs)?

Idea: Use a neural network to \textit{implicitly} represent 3D volume!
NeRF scene representation

$$(x, y, z, \theta, \phi) \rightarrow F_{\Omega} \rightarrow (r, g, b, \sigma)$$

- Spatial location
- Viewing direction
- Output color
- Output density

$F_{\Omega}$ is a fully-connected neural network with 9 layers and 256 channels.
Differentiable Loss Function

\[
\min_{\Theta} \sum_i ||\text{render}_i(F_{\Theta}) - I_{i}^{gt}||^2
\]
What is the render() function?
How is it differentiable?
Volume rendering model

What is the color from this viewpoint?
Volume rendering model

Rendering model for ray \( r(t) = o + td \):

\[
C \approx \sum_{i=1}^{N} T_i \alpha_i c_i
\]

How much light is blocked earlier along ray:

\[
T_i = \prod_{j=1}^{i-1} (1 - \alpha_j)
\]

How much light is contributed by ray segment \( i \):

\[
\alpha_i = 1 - e^{-\sigma_i \delta t_i}
\]
Numerical integration step easily differentiable

Rendering model for ray \( r(t) = o + td \):

\[
C \approx \sum_{i=1}^{N} T_i \alpha_i c_i
\]

- \( C \): Output color
- \( T_i \): Weight
- \( \alpha_i \): Transmittance
- \( c_i \): Color

Differentiable w.r.t color \((r, g, b)\) and volumetric density \(\sigma\) outputs

How much light is blocked earlier along ray:

\[
T_i = \prod_{j=1}^{i-1} (1 - \alpha_j)
\]

How much light is contributed by ray segment \(i\):

\[
\alpha_i = 1 - e^{-\sigma_i \delta t_i}
\]
Results
Novel View Synthesis

Inputs: sparsely sampled images of scene

Outputs: new views of same scene (rendered by our method)
More detailed and consistent than prior work that represents scene as discrete voxel grid

Neural Volumes [Lombardi 2019]  NeRF
Why do we need r,g,b to be a function of viewpoint?
Viewing directions as input

Radiance distribution for point on side of ship

Radiance distribution for point on water's surface
One key trick to make it work ...
Naively passing in position creates blurry images!
Activity!
Let’s say we train a network to memorize an image. How do we make the image look sharper?
Idea: Encode low-dim coordinates to high-dim features

Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains, Tancik et al.
Use positional encoding

\[
\begin{pmatrix}
\sin(x), \cos(x) \\
\sin(2x), \cos(2x) \\
\sin(4x), \cos(4x) \\
\vdots \\
\sin(2^N x), \cos(2^N x)
\end{pmatrix}
\]

Ground truth image  Standard fully-connected net  With “positional encoding”
Lots of extension and applications!
Generalization

Goal: Train a NeRF for arbitrary new scenes with fewer images
Unknown camera poses

Goal: Estimate poses given a trained NeRF

iNeRF: Inverting Neural Radiance Fields for Pose Estimation
Unknown camera poses + Scene

Goal: Simultaneously estimate pose and scene representation
NERF for Grasping

Grasping Singulated Objects
tl;dr

But manipulating objects require 3D reasoning!

PointNet
End-to-end learning for scattered, unordered point data
Unified framework for various tasks

PointNet
mug?
table?
car?
Classification Part Segmentation Semantic Segmentation

What are Neural Radiance Fields (NeRFs)?
Idea: Use a neural network to implicitly represent 3D volume!

✓ No Discretization ✓ Compressible ✓ Differentiable