# Visual AI

CPSC 532R/533R – 2019/2020 Term 2

**Lecture 5.** Representing 3D skeletons and point clouds

Helge Rhodin

# TA for the next two weeks

Raghav Goyal

- PhD student in the Computer Vision lab with Leonid Sigal

- Same time and room as usual
- Yuchi is still available via Piazza and mail

# Overview

- 11 Lectures (Weeks 1 – 6)
  - Introduction
  - Deep learning basics and best practices
  - Network architectures for image processing
  - Representing images and sparse 2D keypoints
  - Representing dense and 3D keypoints
  - Representing geometry and shape
  - Representation learning I (deterministic)
  - Representation learning II (probabilistic)
  - Sequential decision making
  - Unpaired image translation
  - Attention models

- 3x Assignments
  - Playing with pytorch (5% of points)
  - Pose estimation (10% of points)
  - Shape generation (10% of points)

- 1x Project (40 % of points)
  - Project pitch (3 min, week 6)
  - Project presentation (10 min, week 14)
  - Project report (8 pages, April 14)

- 1x Paper presentation (Weeks 8 – 13)
  - Presentation, once per student (25% of points) (20 min + 15 min discussion, week 8-13)
  - Read and review one out of the two papers presented per session (10% of points)

ecotypes

# Project updates

ecotypes

## Killer whale identification

Andrew W Trites

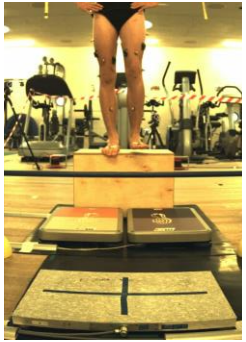
Professor and Director

Institute for the Oceans and Fisheries UBC

*300 images, 40 different whales*
*Sufficient to distinguish ecotypes: transient and residential orcas*

*Sample data available. Drop me a mail if you would like to inspect it.*

## mm-accurate 3D pose and force estimation



Dr. Jörg Spörri

Sport medicine head

University Hospital Balgrist

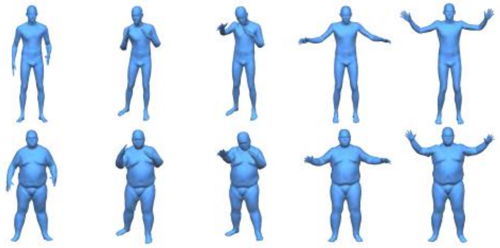
*Pilot: 6 jumps, 2D and 3D pose, pressure plate measurement, video, camera calibration.*
*Final (end of Jan.): 1000 jumps of the same kind*

# Reading: Conditional content generation & Motion transfer

Week 8:

- Park et al., Semantic Image Synthesis with Spatially-Adaptive Normalization
- Li et al., Putting Humans in a Scene: Learning Affordance in 3D Indoor Environments
- Chan et al, Everybody Dance Now
- Gao et al., Automatic Unpaired Shape Deformation Transfer

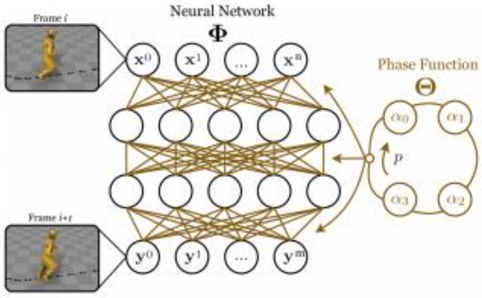


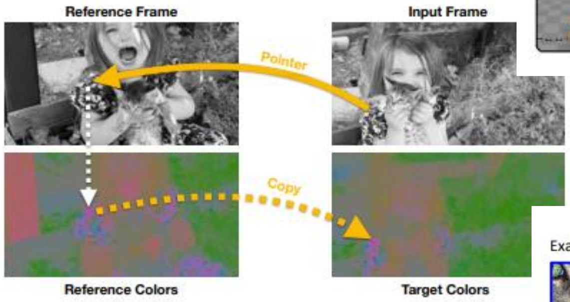


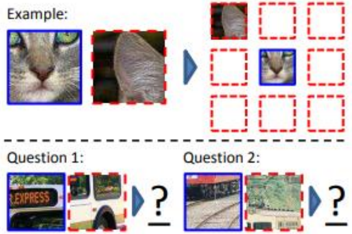
# Reading: Character animation & Self-supervised learning

Week 9:

- Rhodin et al., Interactive Motion Mapping for Real-time Character Control
- Holden et al., Phase-Functioned Neural Networks for Character Control
- Vondrick et al., Tracking Emerges by Colorizing Videos
- Doersch et al., Unsupervised visual representation learning by context prediction



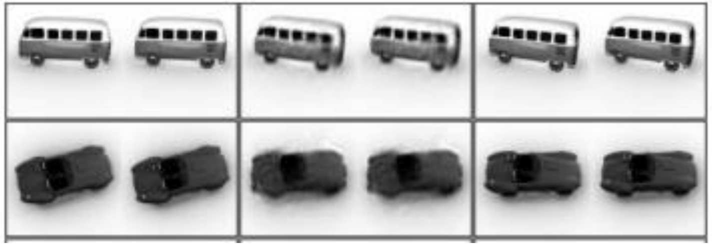



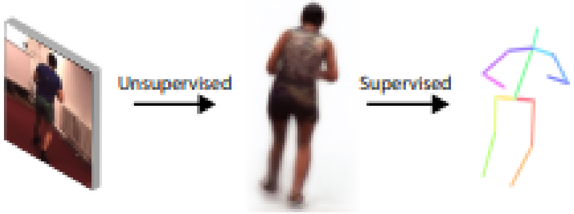
# Reading: Novel view synthesis & Differentiable rendering

Week 10:

- Hinton et al., Transforming Auto-encoders

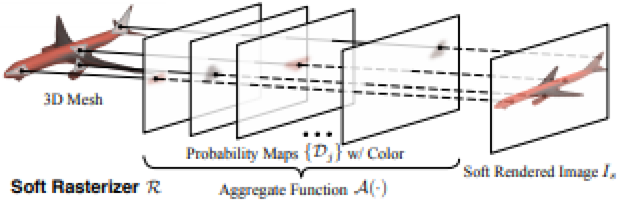


- Rhodin et al., Unsupervised Geometry-Aware Representation for 3D Human Pose Estimation




- Rhodin et al., A Versatile Scene Model with Differentiable Visibility Applied to Generative Pose Estimation

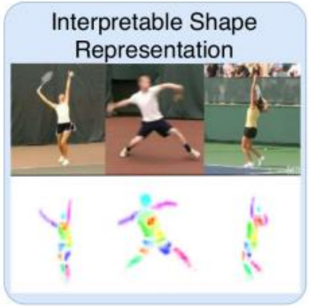
- Liu et al., Soft Rasterizer: A Differentiable Renderer for Image-based 3D Reasoning (changed from preliminary schedule)



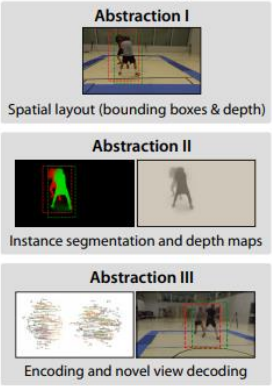
# Reading: Learning person models & Object parts and physics

Week 11:

- Lorenz et al., Unsupervised Part-Based Disentangling of Object Shape and Appearance

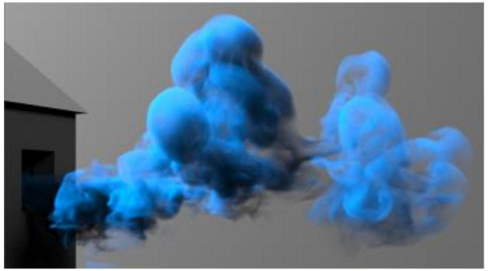



- Rhodin et al., Neural Scene Decomposition for Human Motion Capture




- Li et al., GRASS: Generative Recursive Autoencoders for Shape Structures

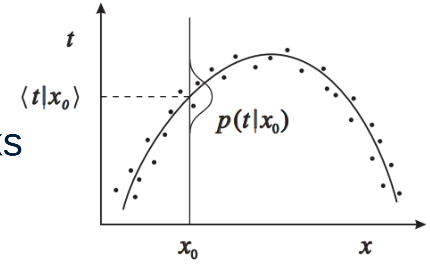
- Xie et al., tempoGAN: A Temporally Coherent, Volumetric GAN for Super-resolution Fluid Flow



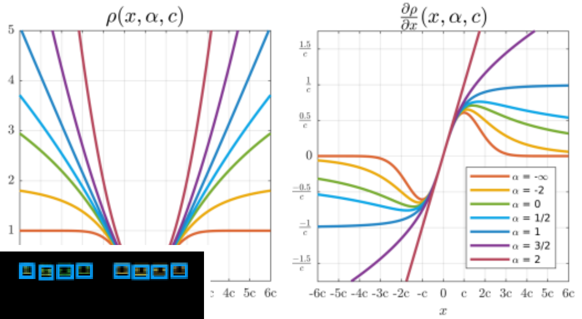
# Reading: Objective functions & Self-supervised object detection

Week 12:

- Christopher Bishop, Mixture Density Networks

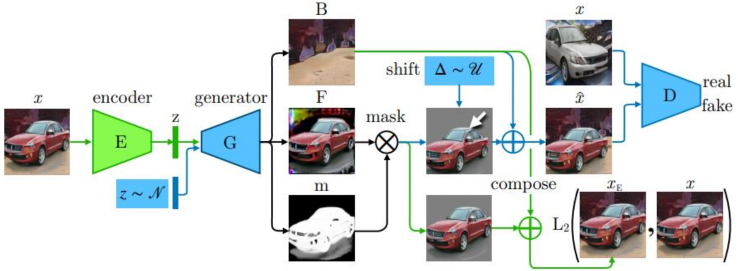



- Jonathan T. Barron, A General and Adaptive Robust Loss Function




- Crawford et al., Spatially invariant unsupervised object detection with convolutional neural networks

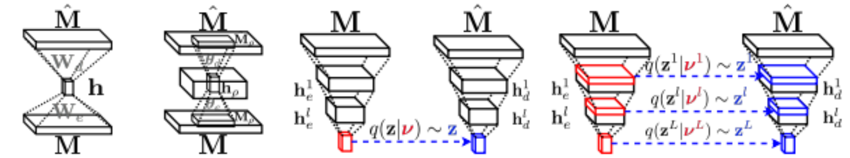
- Bielski and Favaro, Emergence of Object Segmentation in Perturbed Generative Models (changed from preliminary schedule)



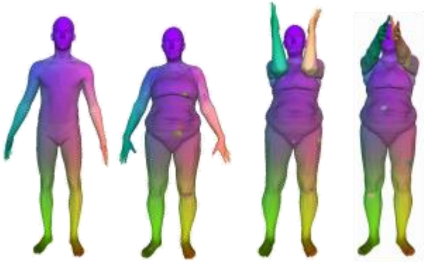
# Reading: Mesh processing & Neural rendering

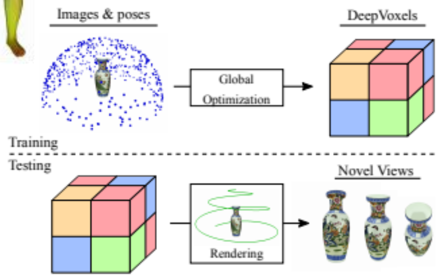
Week 13:

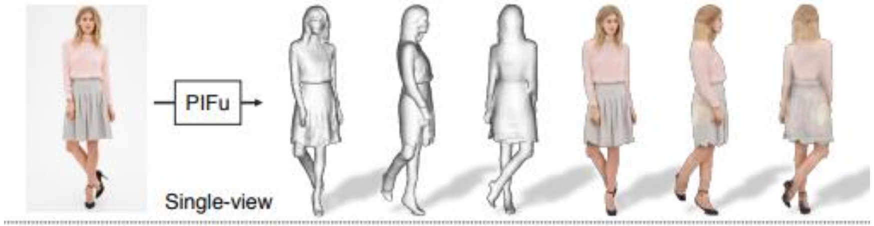
- Bagautdinov et al., Modeling Facial Geometry using Compositional VAEs
- Verma et al., Feastnet: Feature-steered graph convolutions for 3d shape analysis
- Sitzmann et al., DeepVoxels: Learning Persistent 3D Feature Embeddings
- Saito et al., PIFu: Pixel-Aligned Implicit Function for High-Resolution Clothed Human Digitization



(a) Global model (b) Local model (c) Convolutional VAE (d) Compositional VAE





# Assignment clarifications

- **Task III:** Compare the three approaches (regression, classification, integral regression) in terms of the mean squared joint position error. Which of them attains the highest accuracy (lowest error) on the provided validation set? You don't have to train for ages, but make sure that you train all models for the same time. Comment on whether in your setup convergence speed (attaining a decent result early on) or overall accuracy (best result after training all methods for sufficient time) is the main factor.

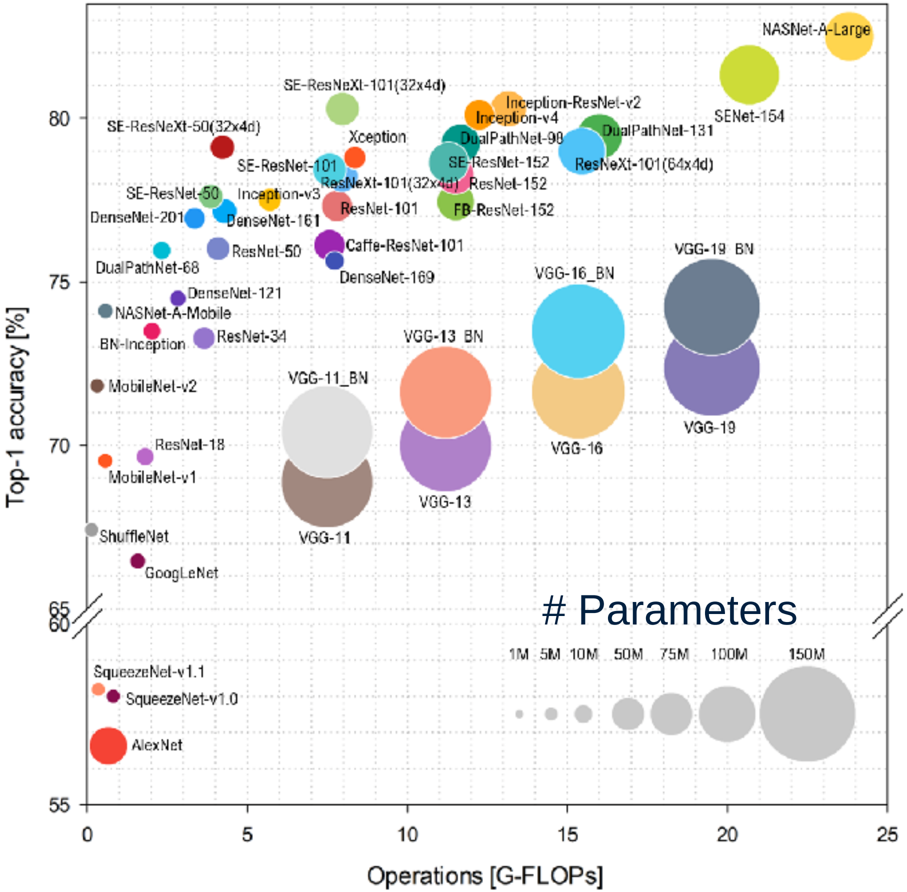
**Submission.** Once finished, submit your jupyter notebook on Canvas. If you have dependencies, add them to a .zip archive. Name your submission `assignment2x_firstName_lastName.ipynb` (or `.zip`).
If some of your outputs are displayed with external tools, such as Tensorboard, please include screenshots of those.

Accessing UBC jupyter servers (slow but easy way)

- > ssh -X rhodin@lin01.students.cs.ubc.ca
- > firefox &

# Recap

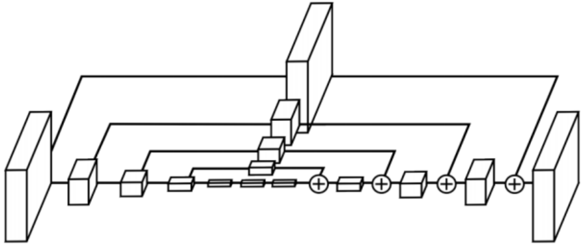
# Recap: Network architectures



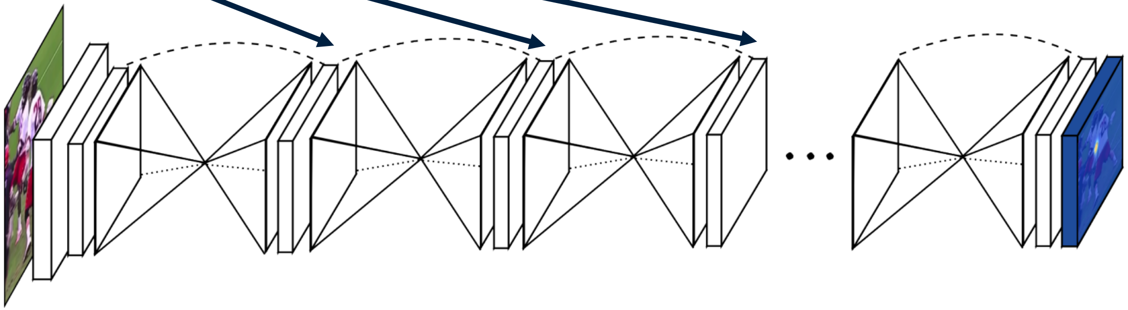
# New: Stacked Hourglass Architecture

Idea: Stacking multiple encoder-decoder networks

- stack of multiple U-Net blocks (usually 2-8)
  - form of iterative refinement
- combined bottom-up (low-level) and top-down (high-level) features
  - encoders: a form of reconstruction (bottom up)
  - decoders: a form of fitting a global model (top-down)
- intermediate supervision (to improve training)



[Newell et al., Stacked Hourglass Networks for Human Pose Estimation]



# Recap: Part affinity fields for associating joints of multiple persons

An extension of heatmaps (positions) to vectors (directions)

- Ground truth affinity field L* between joints *c,k*

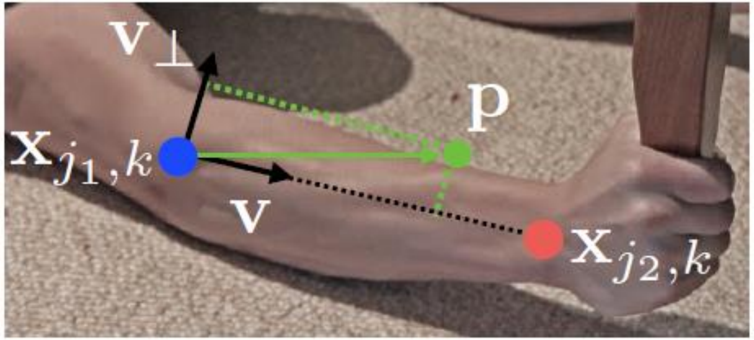
$$\mathbf{L}^*_{c,k}(\mathbf{p}) = \begin{cases} \mathbf{v} & \text{if } \mathbf{p} \text{ on limb } c, k \\ \mathbf{0} & \text{otherwise.} \end{cases}$$

Determine presence by

$$0 \leq \mathbf{v} \cdot (\mathbf{p} - \mathbf{x}_{j_1,k}) \leq l_{c,k} \text{ and } |\mathbf{v}_\perp \cdot (\mathbf{p} - \mathbf{x}_{j_1,k})| \leq \sigma_l,$$

with v defined as

$$\mathbf{v} = (\mathbf{x}_{j_2,k} - \mathbf{x}_{j_1,k}) / ||\mathbf{x}_{j_2,k} - \mathbf{x}_{j_1,k}||_2$$




[Cao et al., Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields, CVPR 2017]

# Recap Integral Regression-based 2D pose estimation

A combination of classification and regression

1. Detection network to produce heatmaps
   - same CNN as for heatmap prediction
2. Soft-max layer to turn heatmap H into probability map P
   - normalizing all pixels in each heatmap H

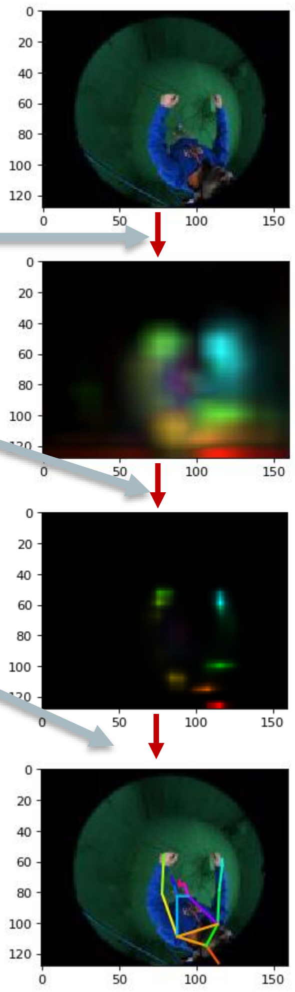
$$P[u,v] = \text{soft-max}(H,(u,v)) = \frac{e^{H[u,v]}}{\sum_{x=1}^{\text{width}} \sum_{y=1}^{\text{height}} e^{H[x,y]}}$$

3. Integration layer to regress joint position (expected position)
   - can be interpreted as voting/weighted average
     *each pixel votes for its own position, weighted by its probability*

$$\text{pose}_x = \sum_{x=1}^{\text{width}} \sum_{y=1}^{\text{height}} x P[x,y]$$

$$\text{pose}_y = \sum_{x=1}^{\text{width}} \sum_{y=1}^{\text{height}} y P[x,y]$$

[Sun et al., Integral Human Pose Regression.]



input

heatmap

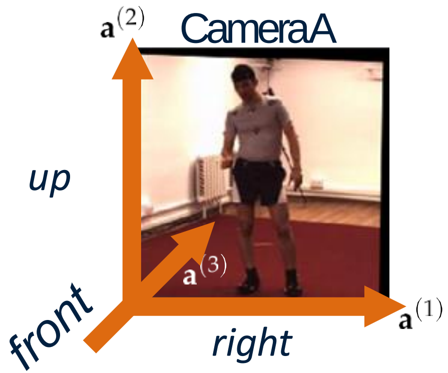
prob. map

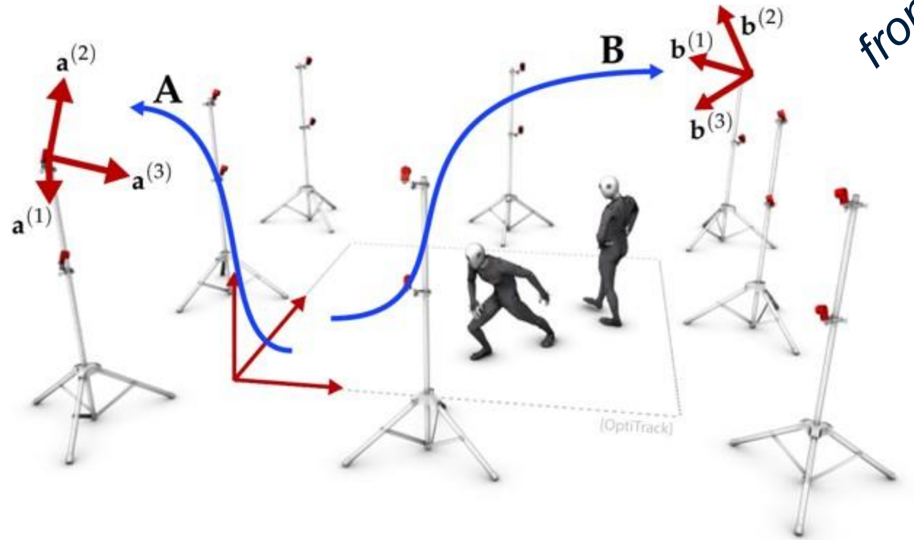
pose vector

# 3D transformations

# Linear transformations

- Used in computer graphics and computer vision
- A chain of linear maps is a linear map
- Rotation
- Scaling
- Shear and mirror



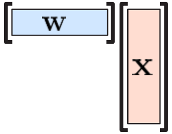



# Affine transformations & augmented matrix and vector

- Can express rigid transformations
  - **Translation**
  - Scale
  - Rotation
  - Shear and mirror

## Linear

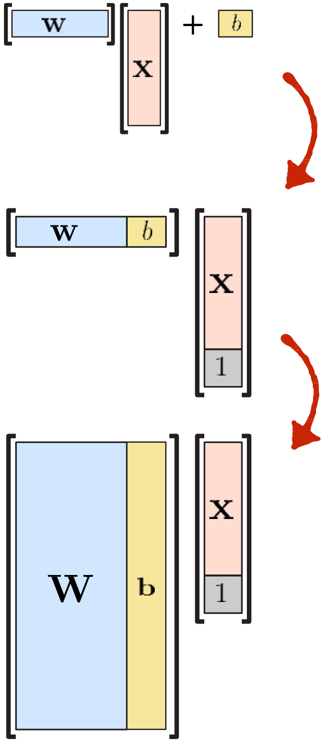
$$f(\mathbf{x}) = \sum_i \mathbf{w}_i \mathbf{x}_i = \mathbf{w} \cdot \mathbf{x}$$




$$f(\mathbf{x}) = \sum_i \mathbf{w}_i \mathbf{x}_i + b = \mathbf{w} \cdot \mathbf{x} + b = \tilde{\mathbf{w}} \cdot \tilde{\mathbf{x}}$$

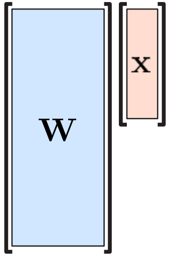
with $\tilde{\mathbf{w}} = (\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_n, b)$
and $\tilde{\mathbf{x}} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n, 1)$

## Affine




## Multidimensional

$$f(\mathbf{x}) = \mathbf{W}\mathbf{x}$$




$$f(\mathbf{x}) = \tilde{\mathbf{W}} \cdot \tilde{\mathbf{x}}$$

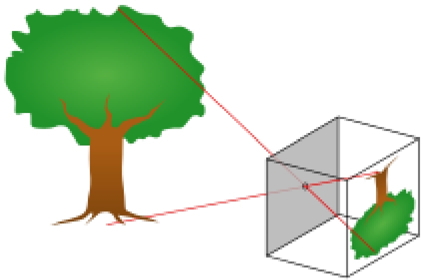
with $\tilde{\mathbf{W}} = \begin{pmatrix} \mathbf{w}_{1,1} & \mathbf{w}_{1,2} & \cdots & \mathbf{w}_{1,n} & b_1 \\ \mathbf{w}_{2,1} & \mathbf{w}_{2,2} & \cdots & \mathbf{w}_{2,n} & b_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \end{pmatrix}$

# Projective transformation & Homogeneous coordinates

Equivalence in homogeneous coordinates

- Compared to the Euclidean space, points are not unique:

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{m-1} \\ x_m \end{bmatrix} = \begin{bmatrix} x_1\lambda \\ x_2\lambda \\ \vdots \\ x_{m-1}\lambda \\ x_m\lambda \end{bmatrix} = \begin{bmatrix} x_1/x_m \\ x_2/x_m \\ \vdots \\ x_{m-1}/x_m \\ 1 \end{bmatrix}$$



Pinhole camera model

[https://en.wikipedia.org/wiki/Pinhole_camera_model]

- Able to model perspective transformations (projection) as a linear transformation

$$\begin{pmatrix} y_1 \\ y_2 \\ 1 \end{pmatrix} \sim \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ 1 \end{pmatrix}$$

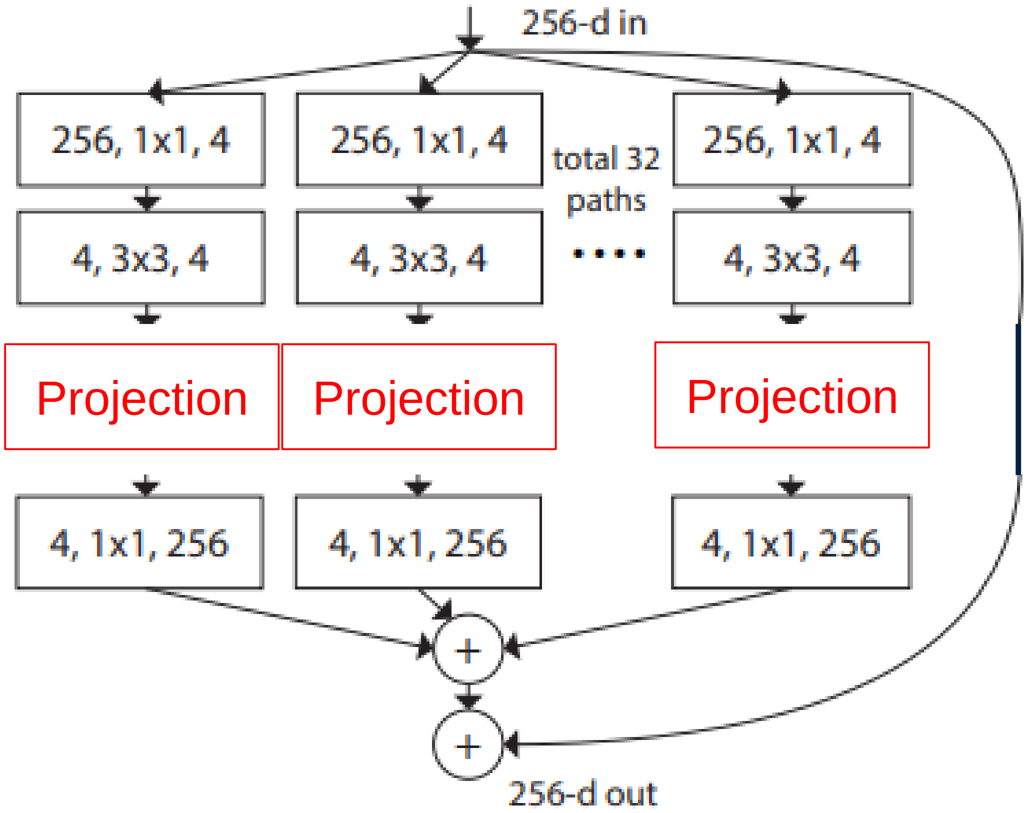
Projection in Homogeneous coordinates

$$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = -\frac{f}{x_3} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

Projection in Euclidean coordinates

# Project Idea: Projective transformations within CNNs (ProjResNext)

- The basis building block of NNs are affine transformations (linear + bias)
- Idea: Use projective transformations instead
- Tasks:
  - Literature review, has this been tried?
  - How to initialize (to prevent vanishing gradients)
  - Do we need to adapt other NN structures, e.g., Batch Norm?
  - Will it be better?



# 3D representations

# Depth maps

Representation: a depth value per pixel

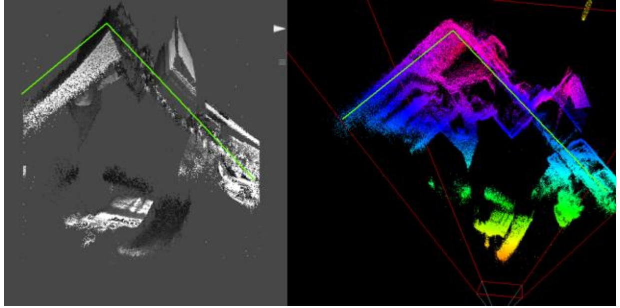
- Size: W x H (Width x Height)
- A 2.5 D representation
    - Continuous in Z (depth)
    - Discrete in X,Y (horizontal and vertical)

Use cases

- Monocular and stereo reconstruction
- Novel view synthesis
- Well-suited for 2D convolution operations

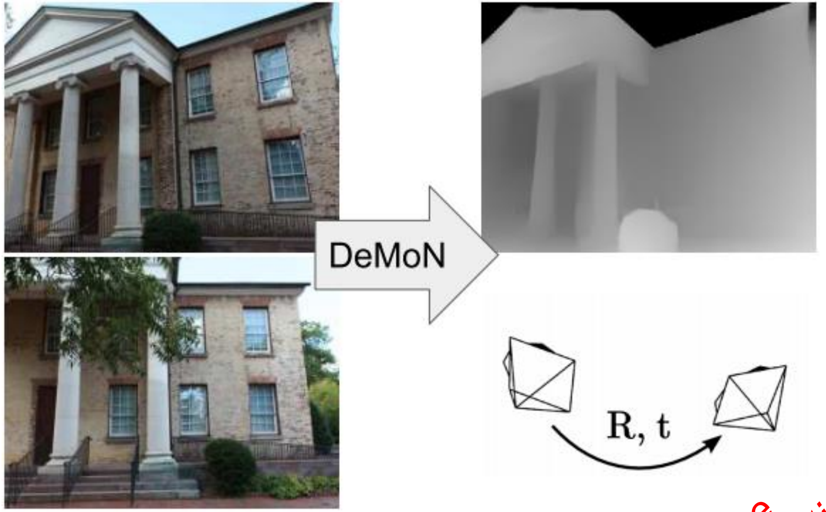
Drawbacks

- Missing parts and holes
- No semantics/correspondence between frames



Kinect depth map viewed from the top

https://stackoverflow.com/questions/37198974/microsoft-kinect-v2-unity-3d-depth-warping




[Ummenhofer et al. DeMoN: Depth and Motion Network for Learning Monocular Stereo]

affine transformation

# Point cloud

Representation: A collection of 3D points

- Size: N x D (Number of points, space dimension)
- Sparse 3 D locations (usually, can be in a higher-dimensional)
  - Continuous and adaptive detail

Benefits

- Well suited for structure from motion form keypoints
- Compact representation of sparse keypoint locations
  - human joints, object edges, …
- Ordered point clouds carry semantics (e.g., first point is the head, the second the neck position)

Drawbacks

- Unstructured, not well suited for convolutions etc.
- No orientation information

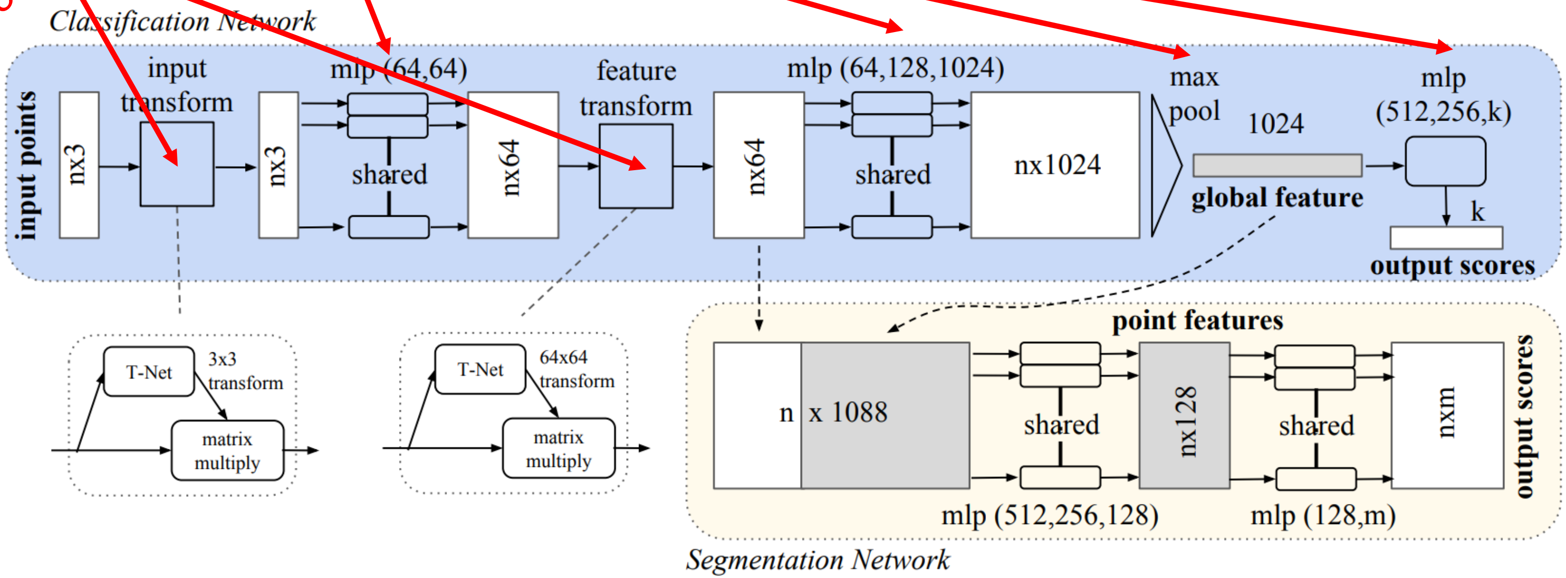
**[Snavely et al., Photo Tourism: Exploring Photo Collections in 3D]**

# PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation

A network architecture to make point cloud processing invariant to

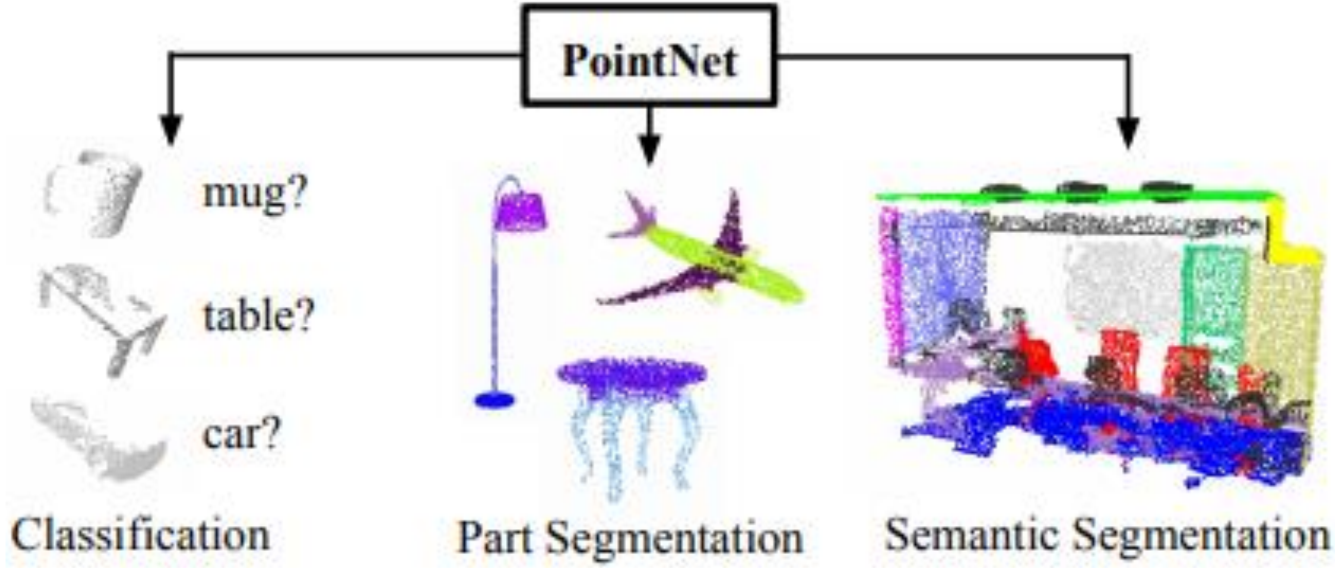
- the point cloud order
- global rigid transform.

affine transformation

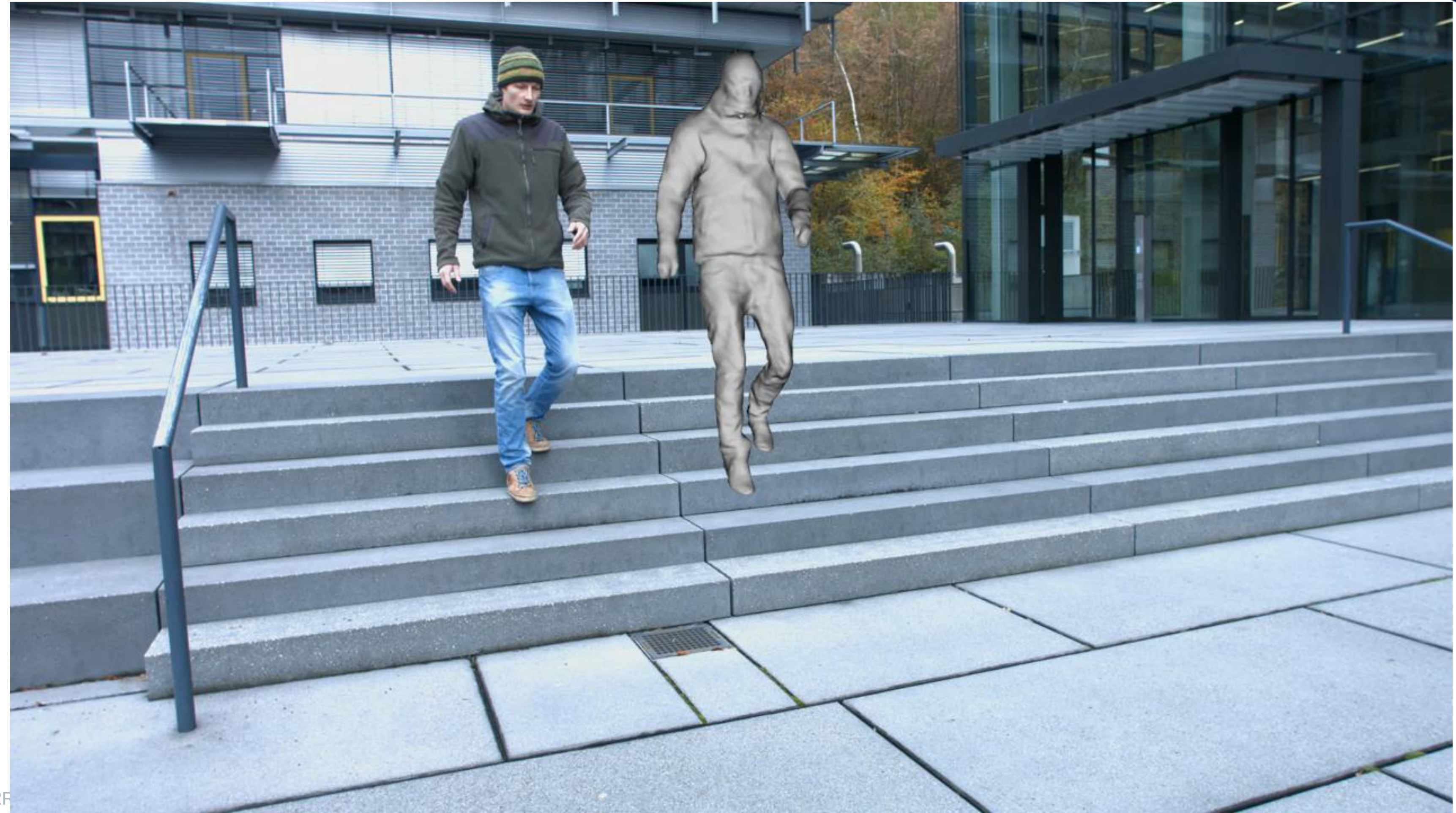


# PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation

Applications




# MonoPerfCap: Human Performance Capture from Monocular Video



# Skeleton representation

Representation: Bones connected by rotational joints

Size: J x (3+1) + B x 1 (# joints, axis + angle, # bones)

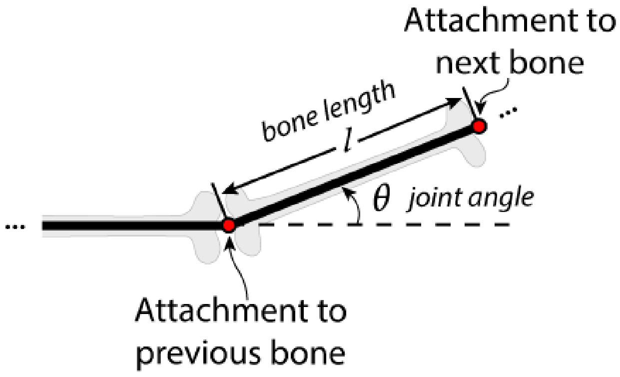
- A hierarchical skeleton approximating anthropology
- Joint rotation is modelled by axis+angle (3 DOF), exponential maps (3-4 DOF) , quaternions (4 DOF) and euler angles (3 DOF)

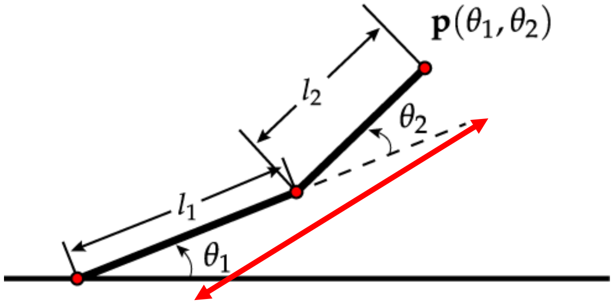
Benefits

- Common for human and animal motion capture
- Enforces skeleton constraints explicitly
- Is efficient to optimize (human tree/star skeleton structure)

Drawbacks

- Only approximates the human skeleton
  (e.g., the shoulder joint is complex to model properly)
- Indirect representation
  - the end effector position depends on all parent joints



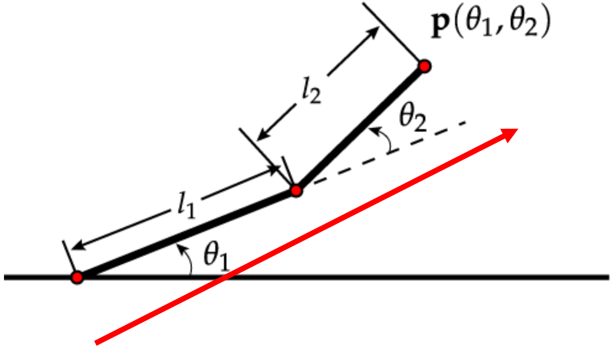




Forward and inverse kinematics

# Forward and inverse kinematics

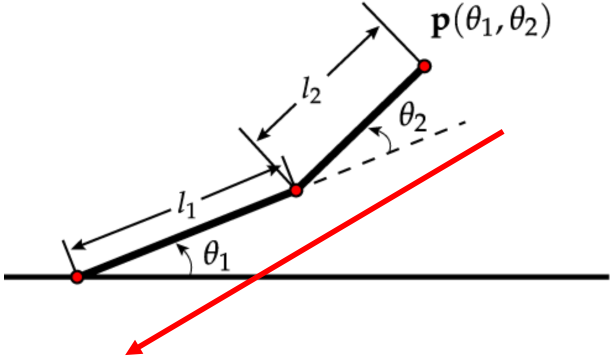
Forward kinematics

- given joint axis, angle, and skeleton hierarchy
- compute joint locations
    - start at the root (neck or head)
    - iteratively continue from parent to child
    - until end-effector is reached
- *a chain of affine transformations!*




Inverse kinematics

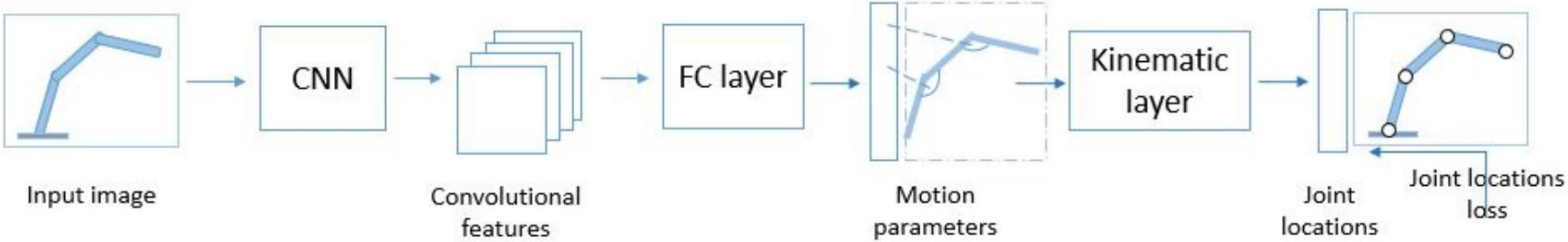
- given skeleton hierarchy and goal location
- optimize joint angles
    - iteratively,  gradient descent (as for NNs)
- minimize distance between end effector (computed by forward kinematics) and goal locations



# Deep Kinematic Pose Regression

Regressing joint angles and bone length instead of joint position

- Change of coordinates enforces prior information
  - bone length symmetry
  - constant bone length (over time)




- Is better than predicting points and enforcing symmetry explicitly
  [Imposing Hard Constraints on Deep Networks: Promises and Limitations]
  - Feasible using Karush-Kuhn-Tucker Conditions
  - Did not work well in practice

Positively Negative

*Workshop on Negative Results in Computer Vision. CVPR 2017*

# Objective functions

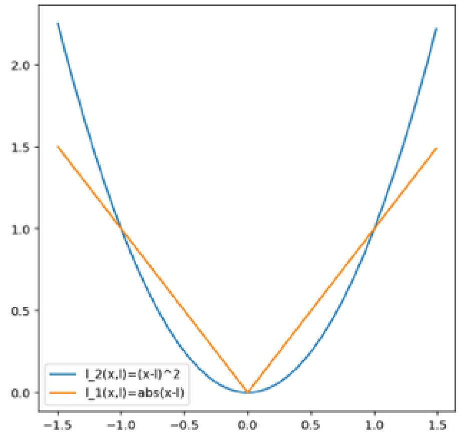
# Recap: MSE, MAE and Cross Entropy

So far:

- simple losses operating element-wise
  - the $l_2$ loss / MSE
  - the $l_1$ loss / MAE
- connecting all elements, but treating them equally
  - soft-max + log-likelihood
  - cross entropy

$$l_{\text{log-likelihood}}(x, y) = -\log(\text{soft-max}(f(x), y))$$

$$l_{\text{cross entropy}}(x, y) = -\sum_{j=1}^{K} y_{[j]} \log(f_{[j]}(x))$$




Quadratic loss

$$l_2(y, l) = (y - l)^2$$

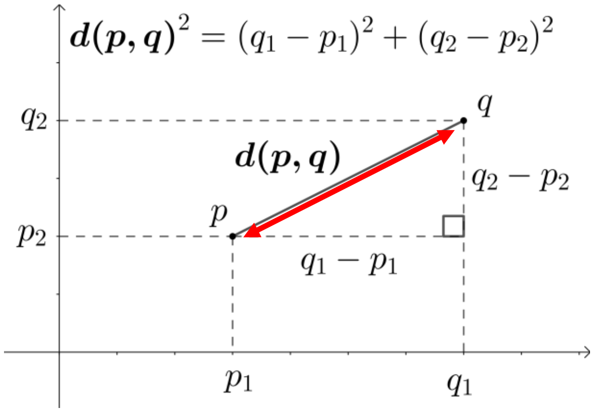
Absolute loss

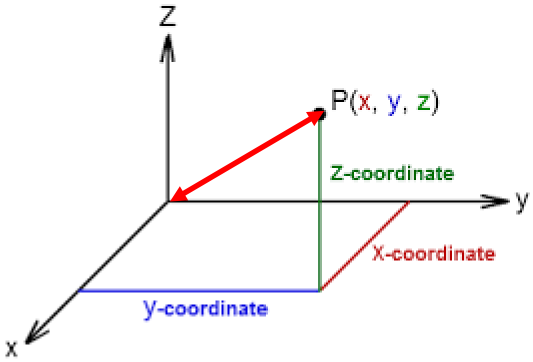
$$l_1(y, l) = |y - l|$$

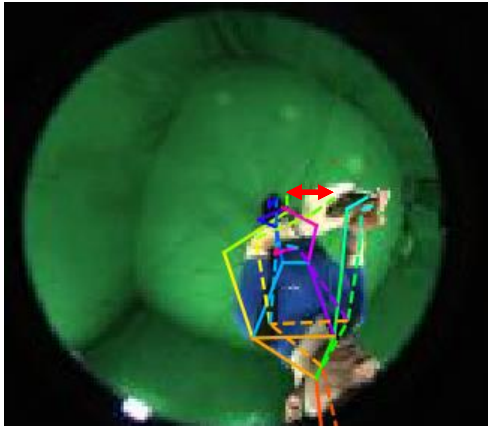
# Mean Per-Joint Position Error (MPJPE)

Euclidean distance

- the square root of the sum of squared coordinate offsets



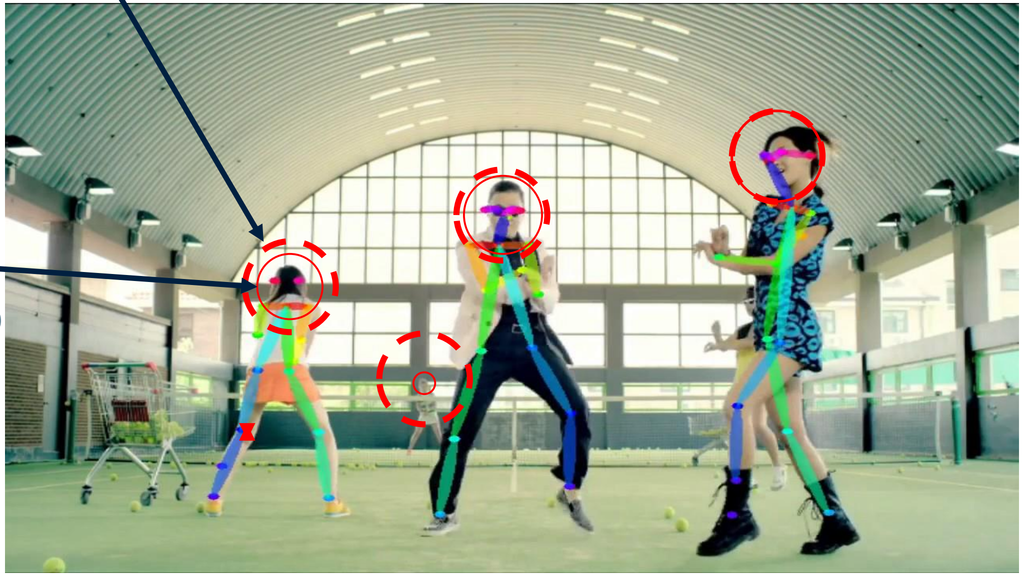
Distance of prediction (solid) to ground truth (dashed)

- averaged over all points
  - groups elements
    - 2D: group of 2 elements, e.g., tensor of N x 18 x 2 for a skeleton with 1
    - 3D: group of 3 elements

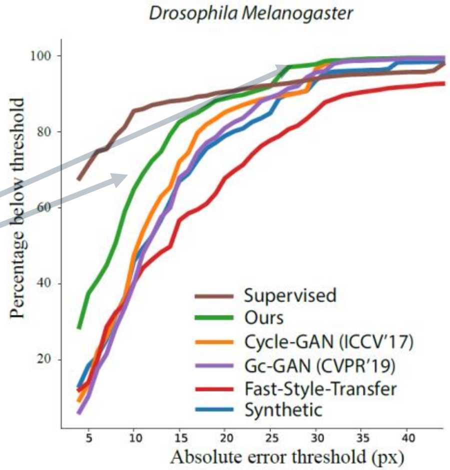
# Percentage of Correct Keypoints (PCK)

- The number of keypoints below a threshold
  - usually using Euclidean distance
  - less sensitive to outliers
  - scale sensitive

- Scale invariant version: PCKh
  - relative to the scale of the GT annotatio
    - e.g. halt the head-neck distance is common for 2D human pose

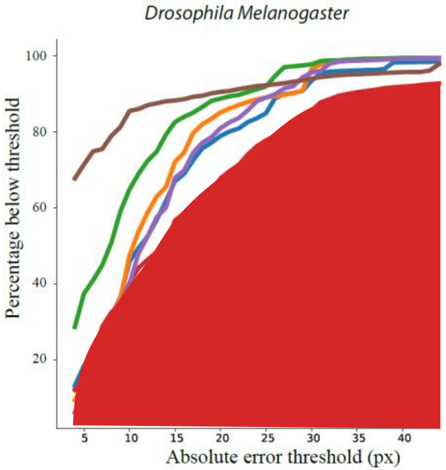


# ROC and AUC




**Receiver operating characteristic (ROC)**

- true positive rate (TPR) against the false positive rate (FPR)
- defined for binary classification
- applicable for any binary metric (e.g., PCK)
- often reveals important details!




**Area Under Curve (AUC)**

- a score for consistency
- the integral (sum) of PCK over different thresholds
- summarizes the ROC curve in single value
  - good for ranking approaches with different precision-recall tradeoffs

# Chamfer distance

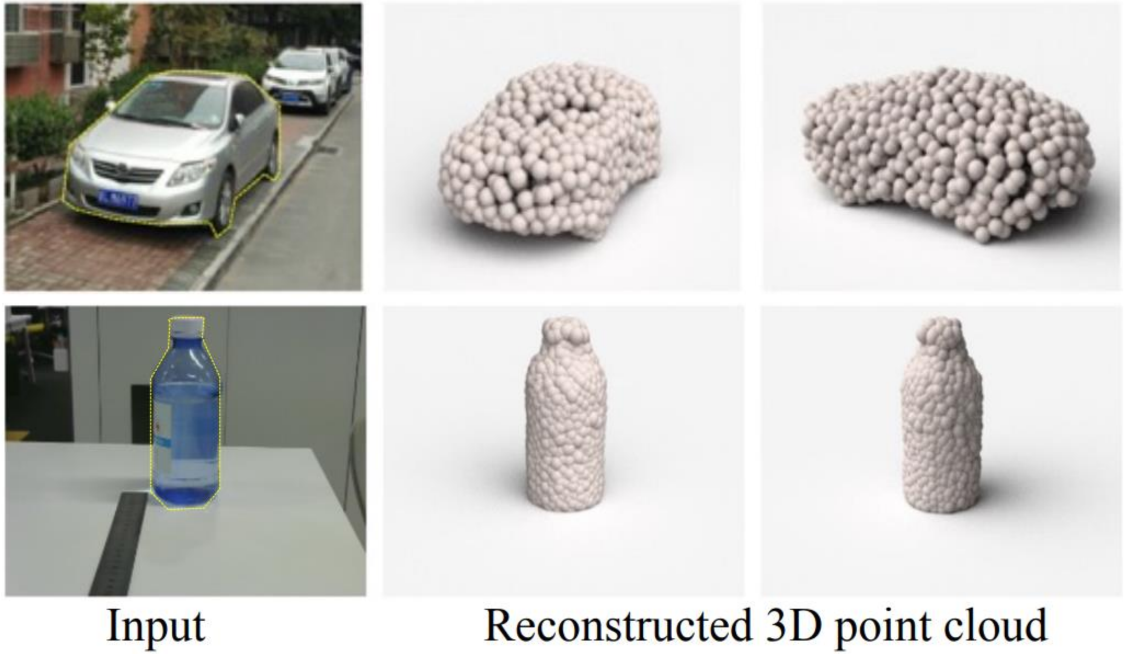
A distance between point clouds without correspondence

- sum of distances between closest points
- bi-directional
  - closest point of y in Y for all x in X
  - closest point of x in X for all y in Y

$$d_{CD}(S_1, S_2) = \sum_{x \in S_1} \min_{y \in S_2} \|x - y\|_2^2 + \sum_{y \in S_2} \min_{x \in S_1} \|x - y\|_2^2$$

  - is not a *distance function* in the mathematical sense, because the triangle inequality does not hold

# A Point Set Generation Network for 3D Object Reconstruction from a Single Image




Shape completion

# Hidden questions

# Surface mesh

Representation: Vertices connected by edges forming faces

- Size: N x D + E x 2 (# points, space dimension, # edges)
- A 3 D surface parametrization (can be in a higher-dimensional)
  - Piece-wise linear with adaptive detail; triangle faces are usual

Benefits

- Good for single and multi-view reconstruction
- Often used for body and object models
- Graph convolutions possible

Drawbacks

- Irregular structure (number of neighbors, edge length, face area)
- Difficult to change topology

  (shape changes require to create new vertices and edges)