Lecture 15

Scene understanding

Bill Freeman, Phillip Isola

The UA & the MIT Ukrainian Folk Dance Ensemble Present:

Nightmarket for Ukraine

Featuring:

Ukrainian food

16 MIT & Berklee performing groups

Games celebrating Ukrainian Culture

Sunday || 4/3 || 7-10 pm

Walker Memorial

BUY TICKETS HERE

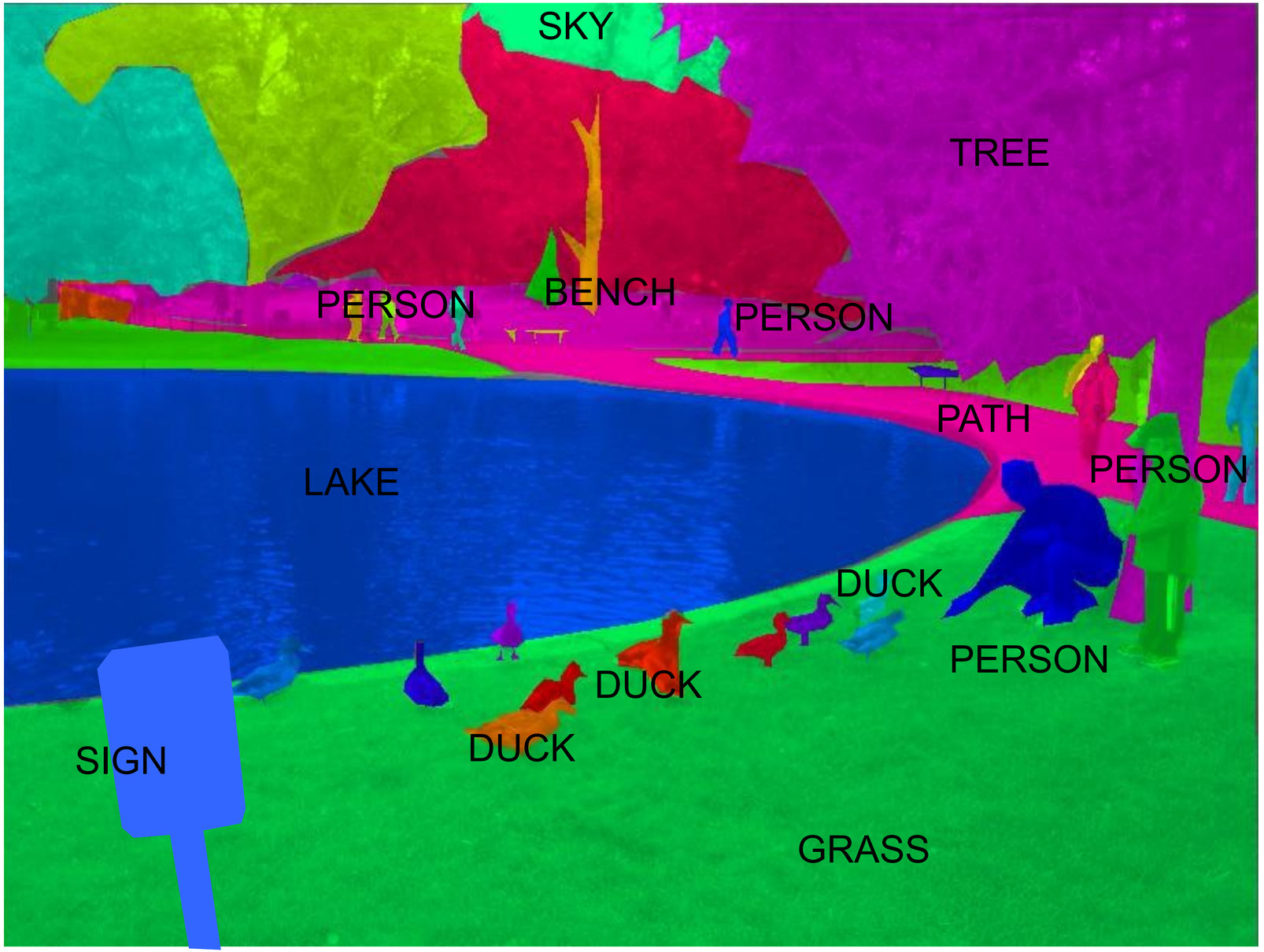
All proceeds will directly support the Ukrainian People

Reminder of pset policies

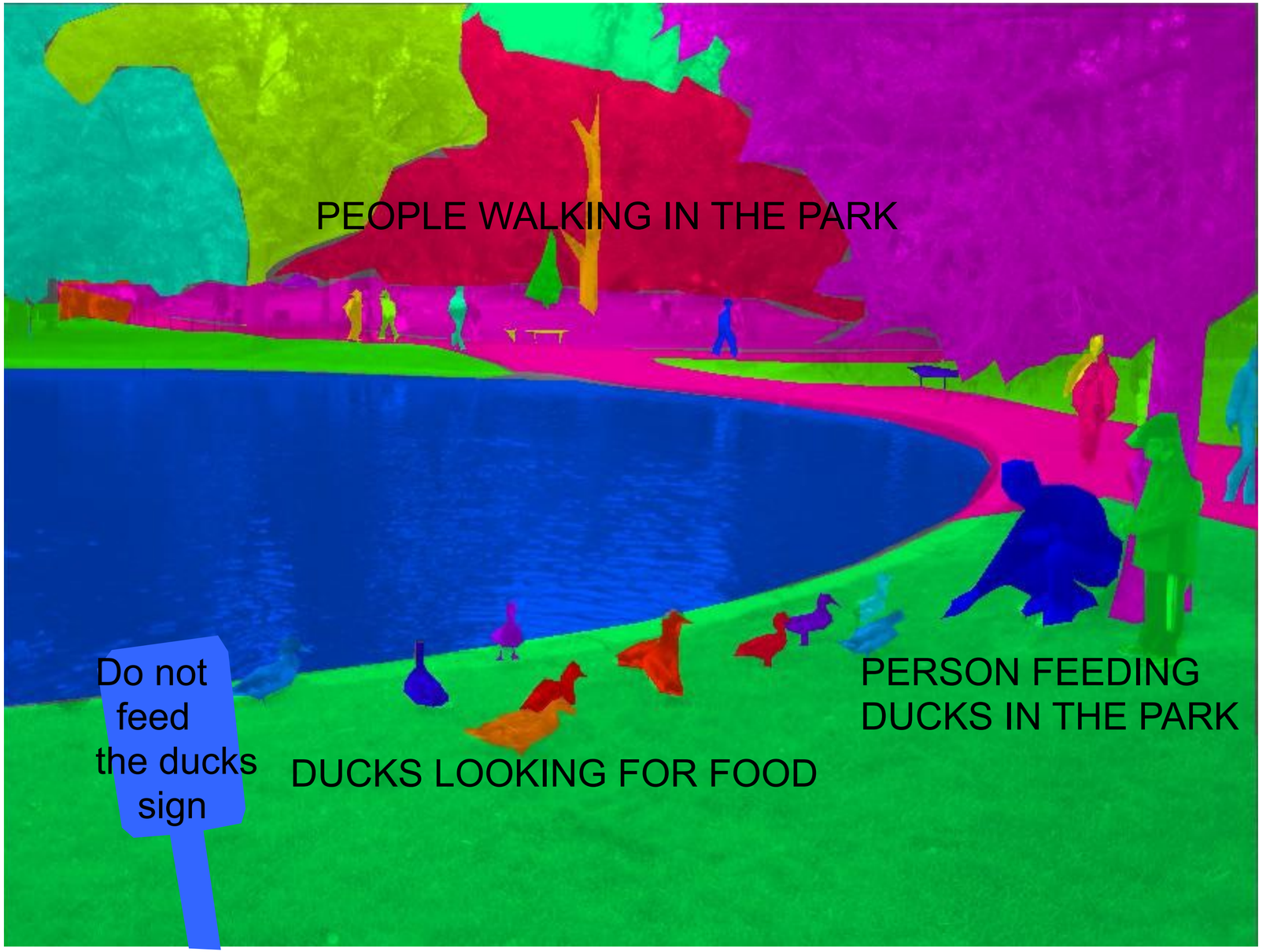
- You should not share answers or look at each others code — you can discuss high level confusions, get advice on how to get started, get help from TAs
- You should not look at solutions from previous years that you may find online
- We do monitor this

15. Scene Understanding

- **Semantics**
  - Object detection
  - Semantic segmentation
  - Instance segmentation
- **Geometry**
  - 3D in the deep learning era
  - Single view depth estimation
  - Unsupervised learning of monocular depth cues

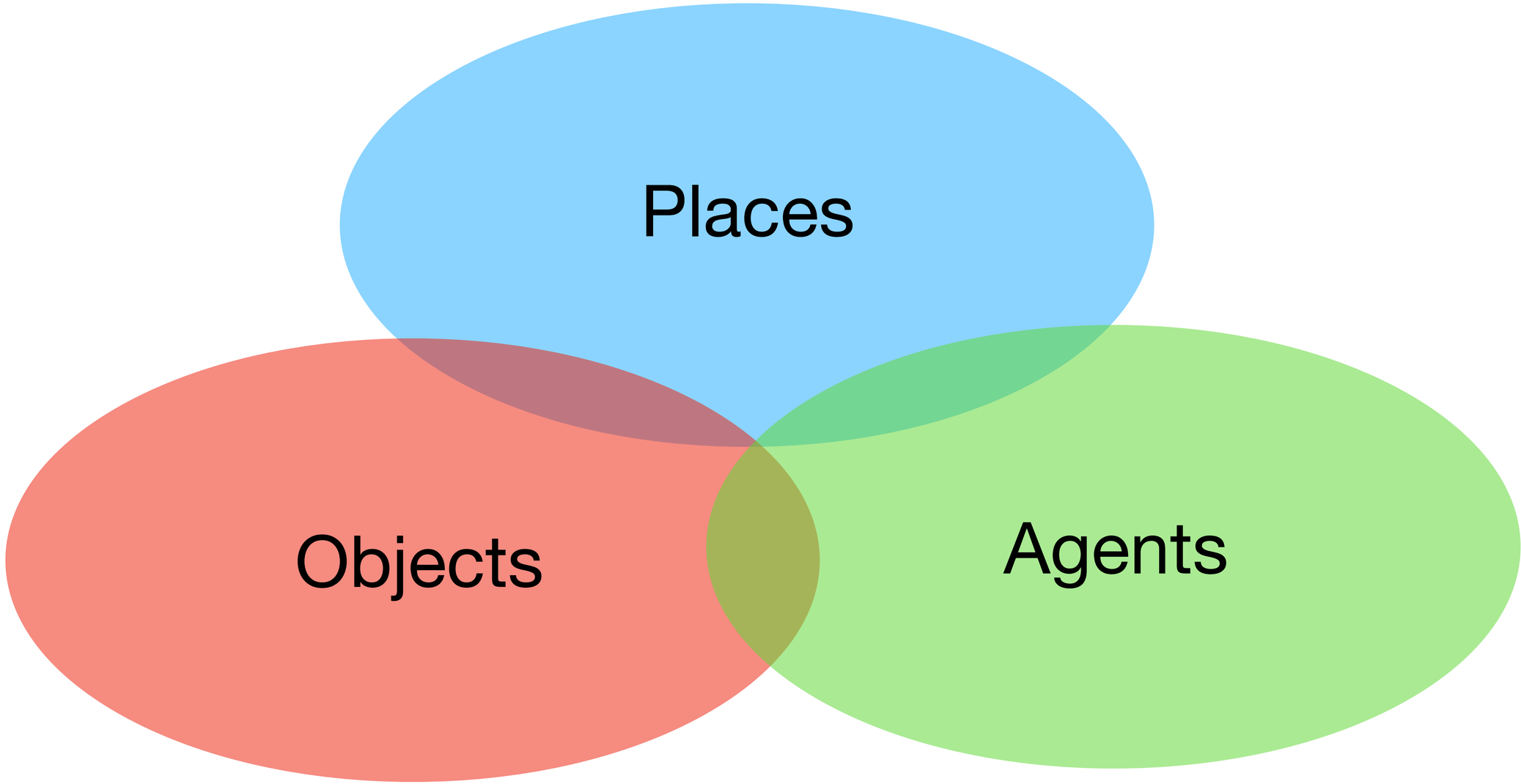

SKY
TREE
PERSON
BENCH
PERSON
PATH
LAKE
PERSON
DUCK
PERSON
DUCK
DUCK
SIGN
GRASS

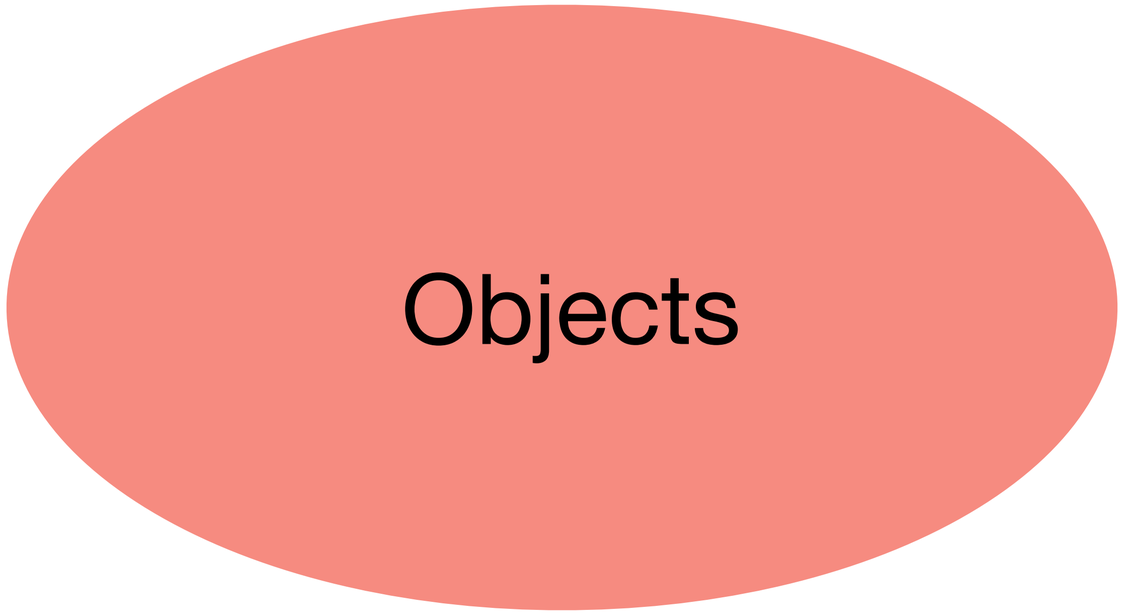

A VIEW OF A PARK ON A NICE SPRING DAY


PEOPLE WALKING IN THE PARK
Do not feed the ducks sign
DUCKS LOOKING FOR FOOD
PERSON FEEDING DUCKS IN THE PARK


PEOPLE UNDER THE
SHADOW OF THE TREES
PLEASE
DO NOT
FEED
THE
DUCKS
DUCKS ON TOP
OF THE GRASS

Scene understanding

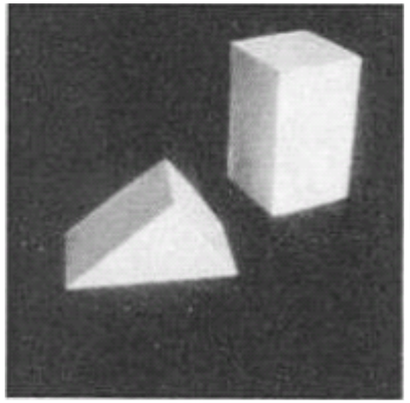



Objects

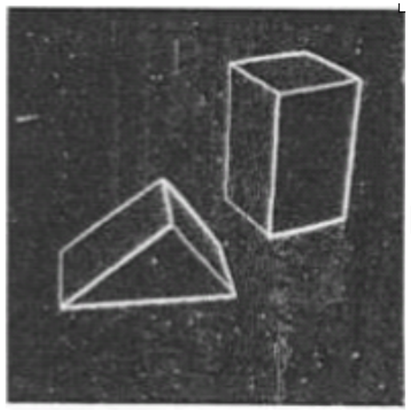
A bit of history…

So, let's make the problem simpler: Block's world

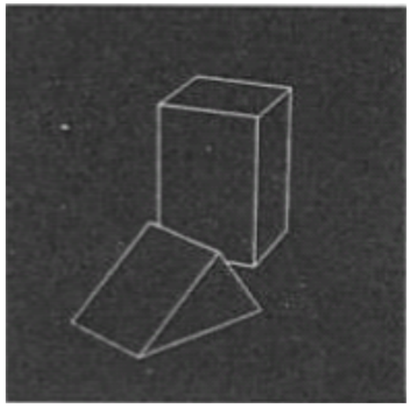
Input

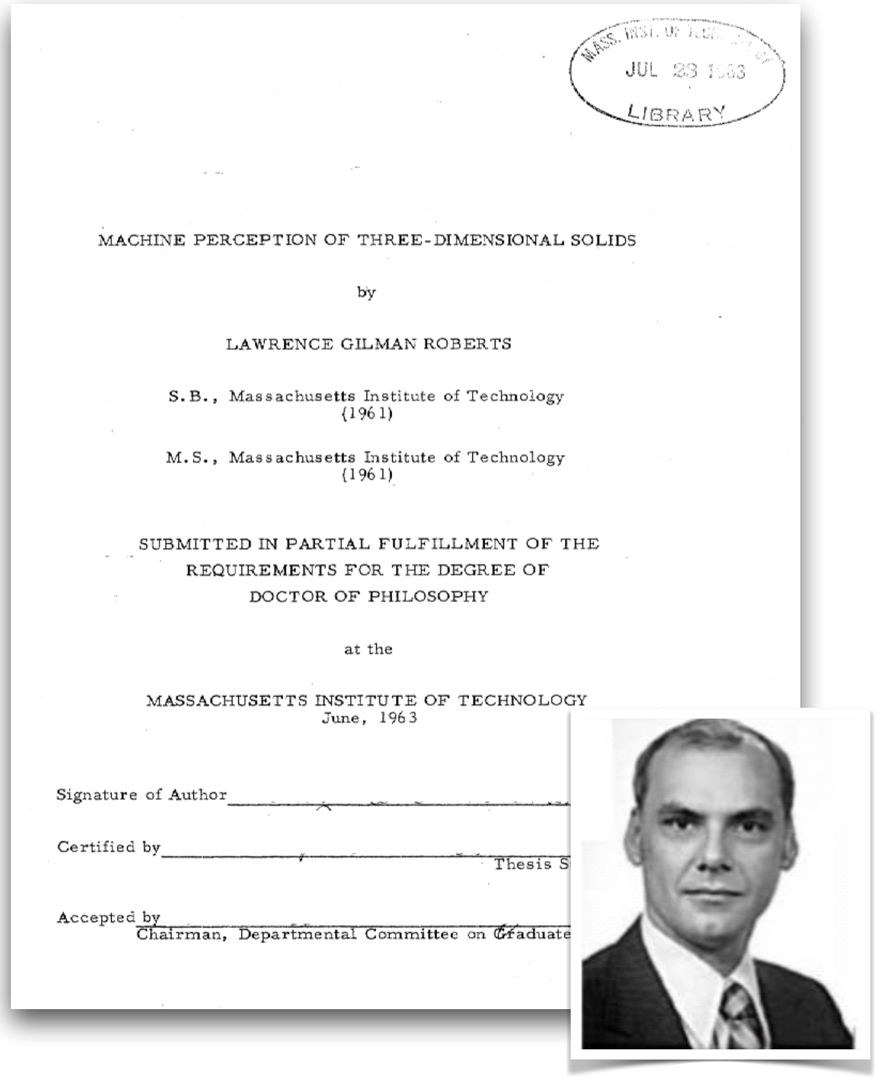


Edges (2x2 gradient)



Reconstructed 3D scene (new view point)





MACHINE PERCEPTION OF THREE-DIMENSIONAL SOLIDS

by

LAWRENCE GILMAN ROBERTS

S.B., Massachusetts Institute of Technology
(1961)

M.S., Massachusetts Institute of Technology
(1961)

SUBMITTED IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY
June, 1963

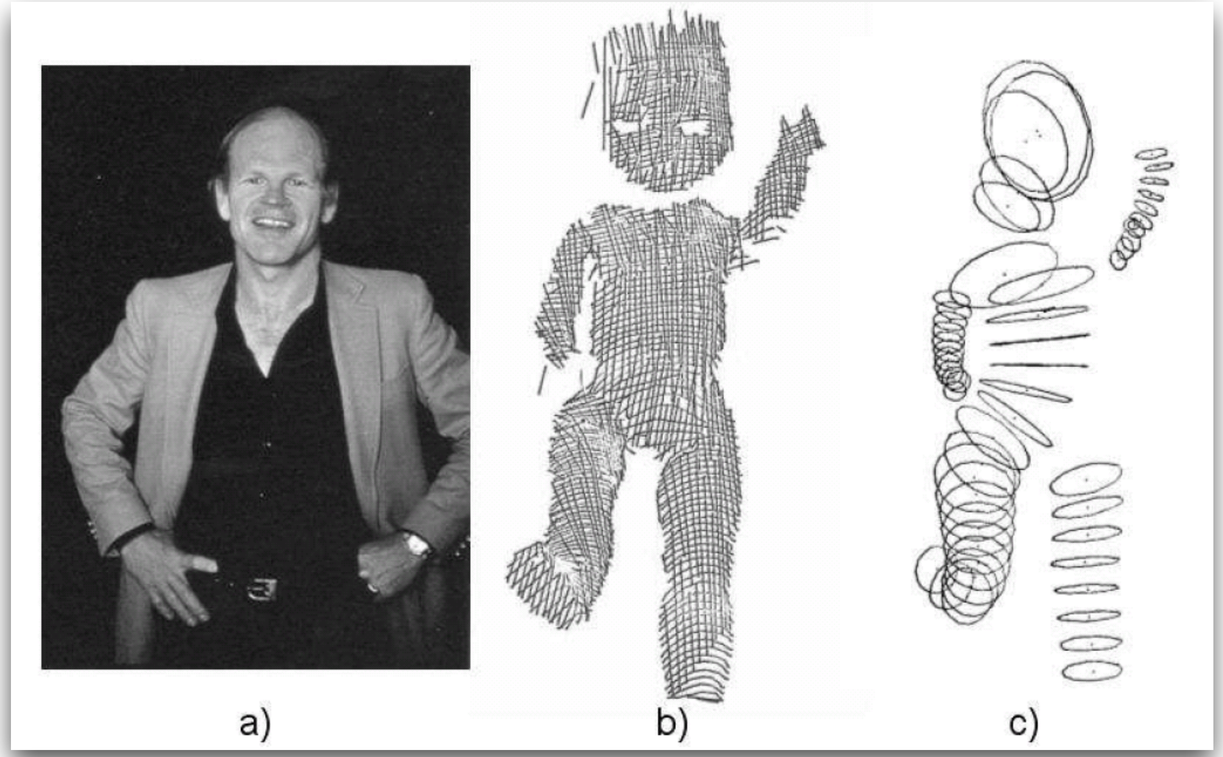
Signature of Author

Certified by
Thesis S

Accepted by
Chairman, Departmental Committee on Graduate

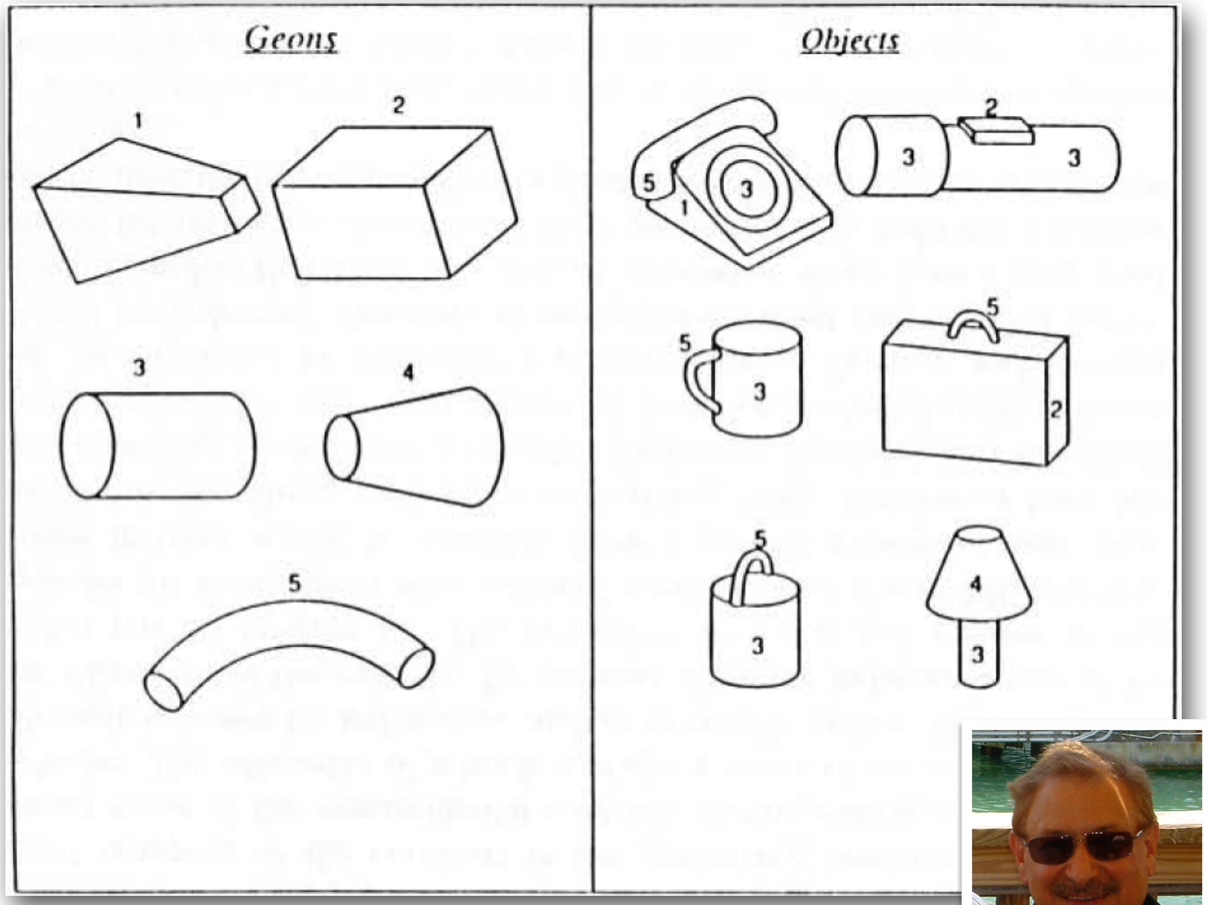
3D, compositional models

Binford and generalized cylinders



Object Recognition in the Geometric Era: a Retrospective. Joseph L. Mundy. 2006

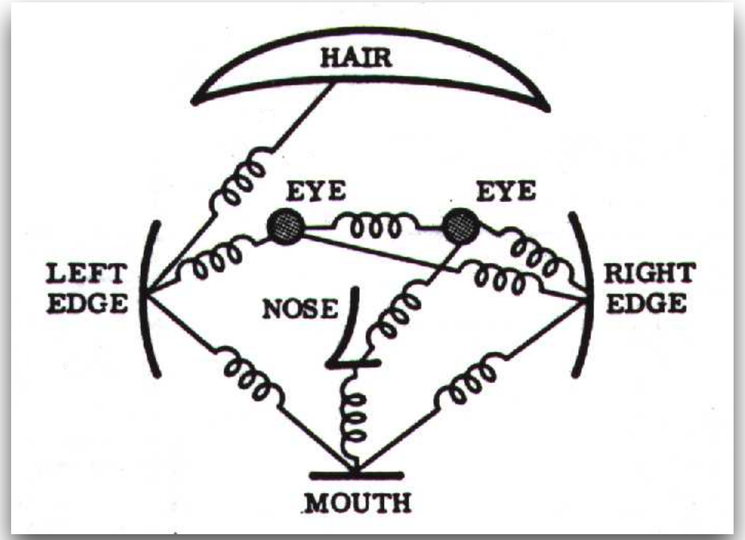
Recognition by components



Recognition-by-Components: A Theory of Human Image Understanding. Psychological Review, 1987.

Irving Biederman

Part based models



- Object as set of parts
  - Generative representation
- Model:
  - Relative locations between parts
  - Appearance of part
- Issues:
  - How to model location
  - How to represent appearance
  - Sparse or dense (pixels or regions)
  - How to handle occlusion/clutter

The Representation and Matching of Pictorial Structures

MARTIN A. FISCHLER AND ROBERT A. ELSCHLAGER

Abstract—The primary problem dealt with in this paper is the following. Given some description of a visual object, find that object in an actual photograph. Part of the solution to this problem is the specification of a descriptive scheme, and a metric on which to base the decision of "goodness" of matching or detection.

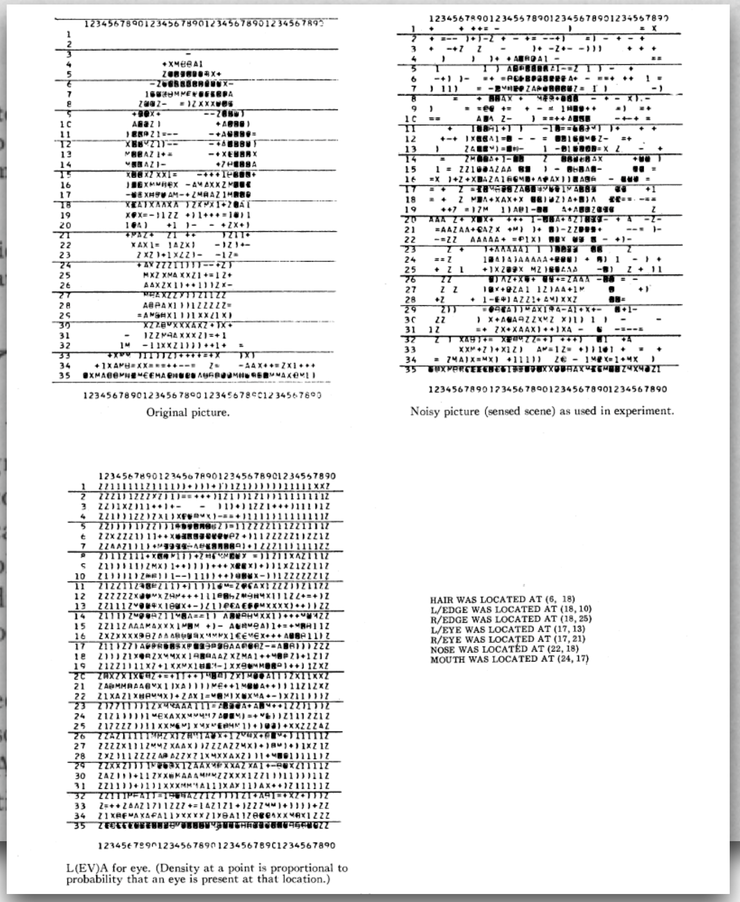
We offer a combined descriptive scheme and decision metric which is general, intuitively satisfying, and which has led to promising experimental results. We also present an algorithm which takes the above descriptions, together with a matrix representing the intensities of the actual photograph, and then finds the described object in the matrix. The algorithm uses a procedure similar to dynamic programming in order to cut down on the vast amount of computation otherwise necessary.

One desirable feature of the approach is its generality. A new programming system does not need to be written for every new description; instead, one just specifies descriptions in terms of a certain set of primitives and parameters.

There are many areas of application: scene analysis and description, map matching for navigation and guidance, optical tracking,

Manuscript received November 30, 1971; revised May 22, 1972, and August 21, 1972.

The authors are with the Lockheed Palo Alto Research Laboratory, Lockheed Missiles & Space Company, Inc., Palo Alto, Calif. 94304.



Original picture.

Noisy picture (sensed scene) as used in experiment.

HAIR WAS LOCATED AT (6, 18)
L/EDGE WAS LOCATED AT (18, 10)
R/EDGE WAS LOCATED AT (18, 25)
L/EYE WAS LOCATED AT (17, 13)
R/EYE WAS LOCATED AT (17, 21)
NOSE WAS LOCATED AT (22, 18)
MOUTH WAS LOCATED AT (24, 17)

L(EV)A for eye. (Density at a point is proportional to probability that an eye is present at that location.)

Scene models

Multiple levels of representation -- pixels > patches > regions > subimages > objects

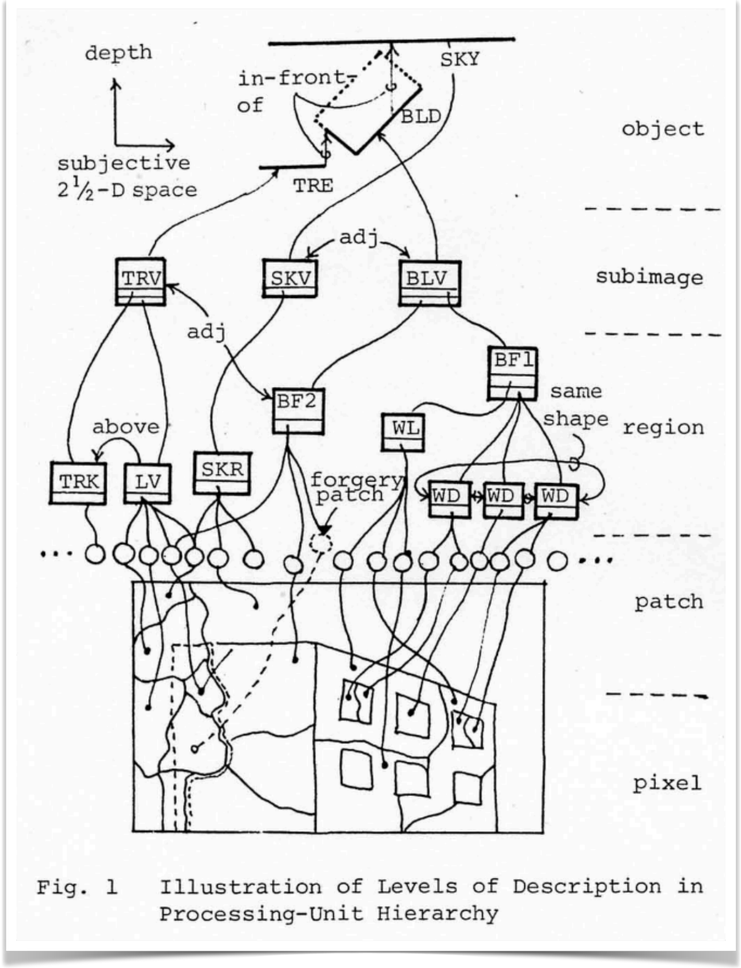
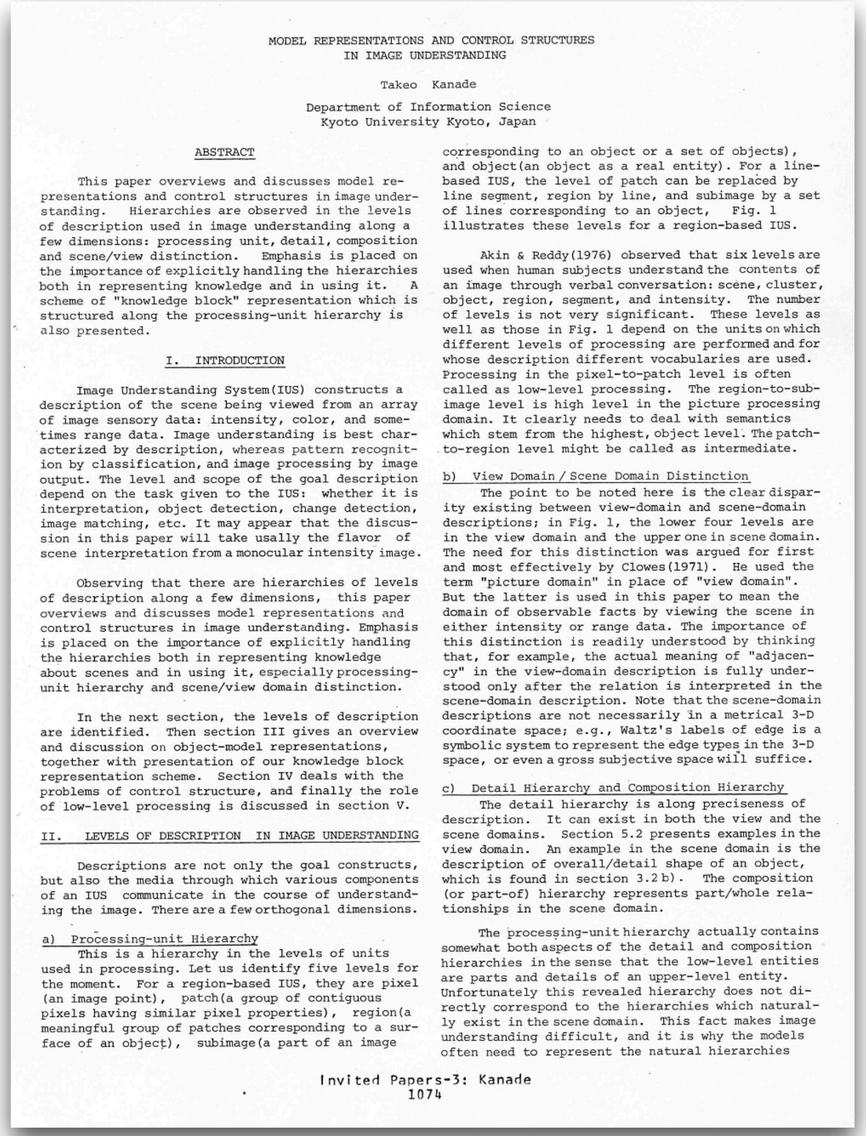


Fig. 1 Illustration of Levels of Description in Processing-Unit Hierarchy



MODEL REPRESENTATIONS AND CONTROL STRUCTURES
IN IMAGE UNDERSTANDING

Takeo Kanade

Department of Information Science
Kyoto University Kyoto, Japan

ABSTRACT

This paper overviews and discusses model representations and control structures in image understanding. Hierarchies are observed in the levels of description used in image understanding along a few dimensions: processing unit, detail, composition and scene/view distinction. Emphasis is placed on the importance of explicitly handling the hierarchies both in representing knowledge and in using it. A scheme of "knowledge block" representation which is structured along the processing-unit hierarchy is also presented.

I. INTRODUCTION

Image Understanding System(IUS) constructs a description of the scene being viewed from an array of image sensory data: intensity, color, and sometimes range data. Image understanding is best characterized by description, whereas pattern recognition by classification, and image processing by image output. The level and scope of the goal description depend on the task given to the IUS: whether it is interpretation, object detection, change detection, image matching, etc. It may appear that the discussion in this paper will take usally the flavor of scene interpretation from a monocular intensity image.

Observing that there are hierarchies of levels of description along a few dimensions, this paper overviews and discusses model representations and control structures in image understanding. Emphasis is placed on the importance of explicitly handling the hierarchies both in representing knowledge about scenes and in using it, especially processing-unit hierarchy and scene/view domain distinction.

In the next section, the levels of description are identified. Then section III gives an overview and discussion on object-model representations, together with presentation of our knowledge block representation scheme. Section IV deals with the problems of control structure, and finally the role of low-level processing is discussed in section V.

II. LEVELS OF DESCRIPTION IN IMAGE UNDERSTANDING

Descriptions are not only the goal constructs, but also the media through which various components of an IUS communicate in the course of understanding the image. There are a few orthogonal dimensions.

a) Processing-unit Hierarchy

This is a hierarchy in the levels of units used in processing. Let us identify five levels for the moment. For a region-based IUS, they are pixel (an image point), patch(a group of contiguous pixels having similar pixel properties), region(a meaningful group of patches corresponding to a surface of an object), subimage(a part of an image corresponding to an object or a set of objects), and object(an object as a real entity). For a line-based IUS, the level of patch can be replaced by line segment, region by line, and subimage by a set of lines corresponding to an object, Fig. 1 illustrates these levels for a region-based IUS.

Akin & Reddy(1976) observed that six levels are used when human subjects understand the contents of an image through verbal conversation: scene, cluster, object, region, segment, and intensity. The number of levels is not very significant. These levels as well as those in Fig. 1 depend on the units on which different levels of processing are performed and for whose description different vocabularies are used. Processing in the pixel-to-patch level is often called as low-level processing. The region-to-subimage level is high level in the picture processing domain. It clearly needs to deal with semantics which stem from the highest, object level. The patch-to-region level might be called as intermediate.

b) View Domain / Scene Domain Distinction

The point to be noted here is the clear disparity existing between view-domain and scene-domain descriptions; in Fig. 1, the lower four levels are in the view domain and the upper one in scene domain. The need for this distinction was argued for first and most effectively by Clowes(1971). He used the term "picture domain" in place of "view domain". But the latter is used in this paper to mean the domain of observable facts by viewing the scene in either intensity or range data. The importance of this distinction is readily understood by thinking that, for example, the actual meaning of "adjacency" in the view-domain description is fully understood only after the relation is interpreted in the scene-domain description. Note that the scene-domain descriptions are not necessarily in a metrical 3-D coordinate space; e.g., Waltz's labels of edge is a symbolic system to represent the edge types in the 3-D space, or even a gross subjective space will suffice.

c) Detail Hierarchy and Composition Hierarchy

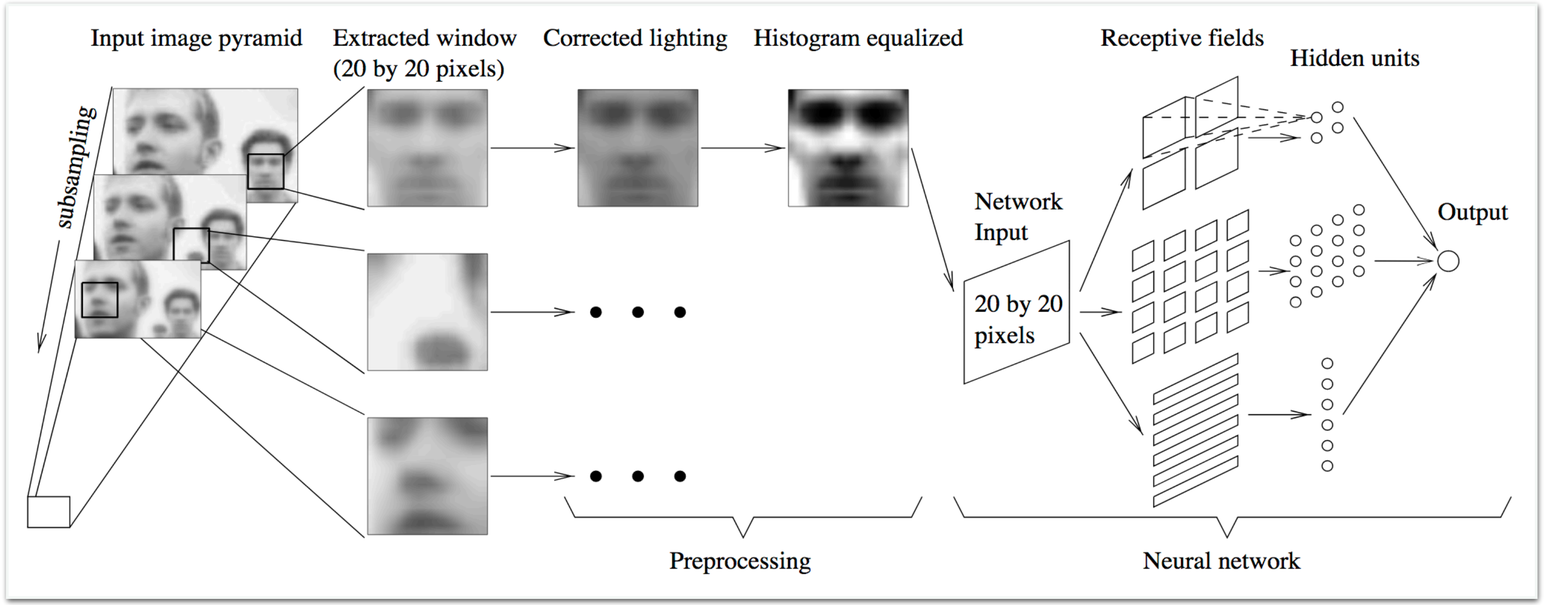
The detail hierarchy is along preciseness of description. It can exist in both the view and the scene domains. Section 5.2 presents examples in the view domain. An example in the scene domain is the description of overall/detail shape of an object, which is found in section 3.2 b). The composition (or part-of) hierarchy represents part/whole relationships in the scene domain.

The processing-unit hierarchy actually contains somewhat both aspects of the detail and composition hierarchies in the sense that the low-level entities are parts and details of an upper-level entity. Unfortunately this revealed hierarchy does not directly correspond to the hierarchies which naturally exist in the scene domain. This fact makes image understanding difficult, and it is why the models often need to represent the natural hierarchies

https://www.ri.cmu.edu/pub_files/pub4/kanade_takeo_1977_1/kanade_takeo_1977_1.pdf

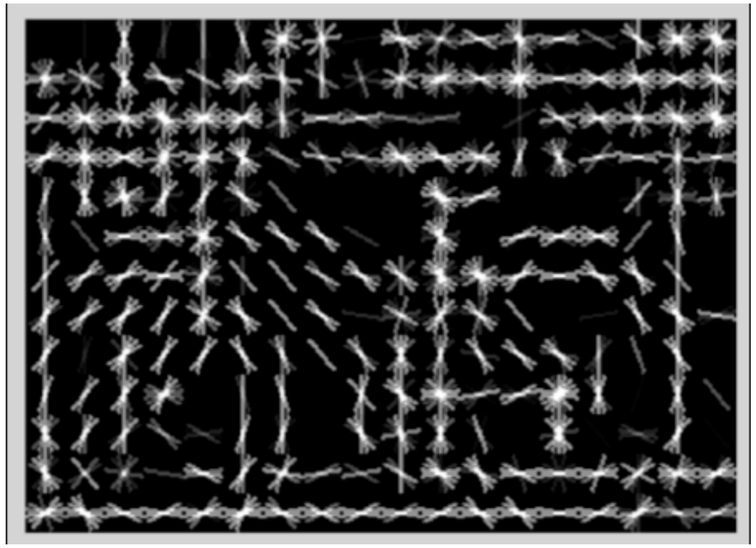
Neural Network-Based Face Detector

Train a set of multilayer perceptrons and arbitrate a decision among all outputs



Rowley, Baluja, and Kanade: Neural Network-Based Face Detection (PAMI, January 1998)

Histograms of oriented gradients (HOG)



1. Bin gradients from 8x8 pixel neighborhoods into 9 orientations
2. Linear SVM

Histograms of Oriented Gradients for Human Detection

Navneet Dalal and Bill Triggs
INRIA Rhône-Alps, 655 avenue de l'Europe, Montbonnot 38334, France
{Navneet.Dalal,Bill.Triggs}@inrialpes.fr, http://lear.inrialpes.fr

Abstract

We study the question of feature sets for robust visual object recognition, adopting linear SVM based human detection as a test case. After reviewing existing edge and gradient based descriptors, we show experimentally that grids of Histograms of Oriented Gradient (HOG) descriptors significantly outperform existing feature sets for human detection. We study the influence of each stage of the computation on performance, concluding that fine-scale gradients, fine orientation binning, relatively coarse spatial binning, and high-quality local contrast normalization in overlapping descriptor blocks are all important for good results. The new approach gives near-perfect separation on the original MIT pedestrian database, so we introduce a more challenging dataset containing over 1800 annotated human images with a large range of pose variations and backgrounds.

1 Introduction

Detecting humans in images is a challenging task owing to their variable appearance and the wide range of poses that they can adopt. The first need is a robust feature set that allows the human form to be discriminated cleanly, even in cluttered backgrounds under difficult illumination. We study the issue of feature sets for human detection, showing that locally normalized Histogram of Oriented Gradient (HOG) descriptors provide excellent performance relative to other existing feature sets including wavelets [17,22]. The proposed descriptors are reminiscent of edge orientation histograms [4,5], SIFT descriptors [12] and shape contexts [1], but they are computed on a dense grid of uniformly spaced cells and they use overlapping local contrast normalizations for improved performance. We make a detailed study of the effects of various implementation choices on detector performance, taking "pedestrian detection" (the detection of mostly visible people in more or less upright poses) as a test case. For simplicity and speed, we use linear SVM as a baseline classifier throughout the study. The new detectors give essentially perfect results on the MIT pedestrian test set [18,17], so we have created a more challenging set containing over 1800 pedestrian images with a large range of poses and backgrounds. Ongoing work suggests that our feature set performs equally well for other shape-based object classes.

We briefly discuss previous work on human detection in §2, give an overview of our method §3, describe our data sets in §4 and give a detailed description and experimental evaluation of each stage of the process in §5–6. The main conclusions are summarized in §7.

2 Previous Work

There is an extensive literature on object detection, but here we mention just a few relevant papers on human detection [18,17,22,16,20]. See [6] for a survey. Papageorgiou *et al* [18] describe a pedestrian detector based on a polynomial SVM using rectified Haar wavelets as input descriptors, with a parts (subwindow) based variant in [17]. Depoortere *et al* give an optimized version of this [2]. Gavrila & Philomen [8] take a more direct approach, extracting edge images and matching them to a set of learned exemplars using chamfer distance. This has been used in a practical real-time pedestrian detection system [7]. Viola *et al* [22] build an efficient moving person detector, using AdaBoost to train a chain of progressively more complex region rejection rules based on Haar-like wavelets and space-time differences. Ronfard *et al* [19] build an articulated body detector by incorporating SVM based limb classifiers over 1st and 2nd order Gaussian filters in a dynamic programming framework similar to those of Felzenszwalb & Huttenlocher [3] and Ioffe & Forsyth [9]. Mikolajczyk *et al* [16] use combinations of orientation-position histograms with binary-thresholded gradient magnitudes to build a parts based method containing detectors for faces, heads, and front and side profiles of upper and lower body parts. In contrast, our detector uses a simpler architecture with a single detection window, but appears to give significantly higher performance on pedestrian images.

3 Overview of the Method

This section gives an overview of our feature extraction chain, which is summarized in fig. 1. Implementation details are postponed until §6. The method is based on evaluating well-normalized local histograms of image gradient orientations in a dense grid. Similar features have seen increasing use over the past decade [4,5,12,15]. The basic idea is that local object appearance and shape can often be characterized rather well by the distribution of local intensity gradients or

1

https://lear.inrialpes.fr/people/triggs/pubs/Dalal-cvpr05.pdf

Families of recognition algorithms

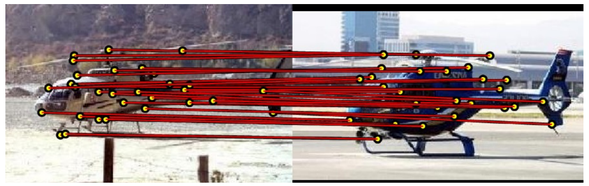
Bag of words models

Csurka, Dance, Fan, Willamowski, and Bray 2004
Sivic, Russell, Freeman, Zisserman,
ICCV 2005

Voting models

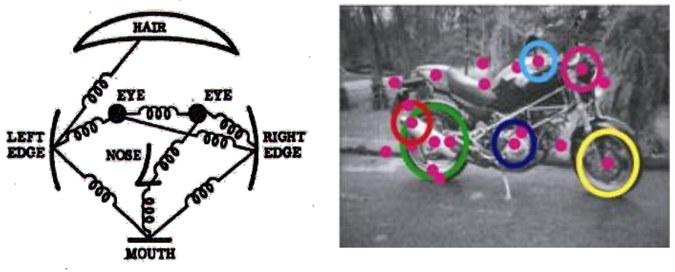
Viola and Jones, ICCV 2001
Heisele, Poggio, et. al., NIPS 01
Schneiderman, Kanade 2004
Vidal-Naquet, Ullman 2003

Shape matching
Deformable models



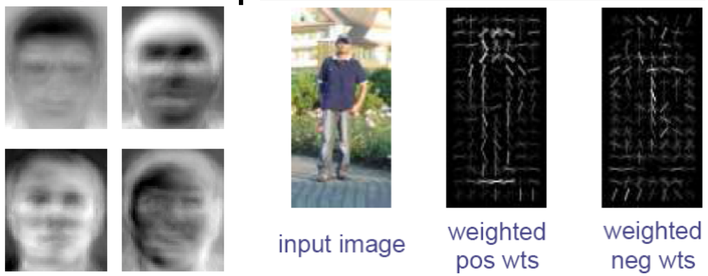
Berg, Berg, Malik, 2005
Cootes, Edwards, Taylor, 2001

Constellation models



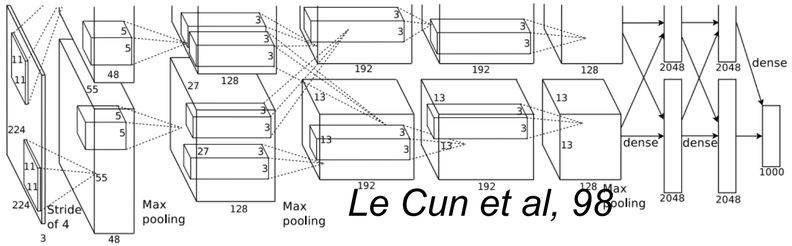
Fischler and Elschlager, 1973
Burl, Leung, and Perona, 1995
Weber, Welling, and Perona, 2000
Fergus, Perona, & Zisserman, CVPR 2003

Rigid template models



Sirovich and Kirby 1987
Turk, Pentland, 1991
Dalal & Triggs, 2006

Neural networks



Le Cun et al, 98


car

Classification

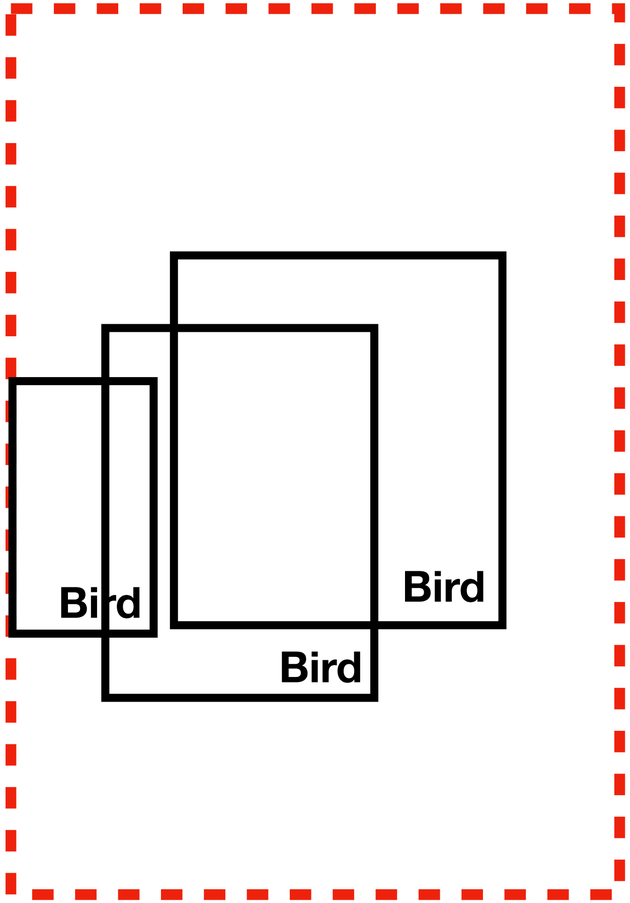
f

Birds

Object detection

f

Classification and localization



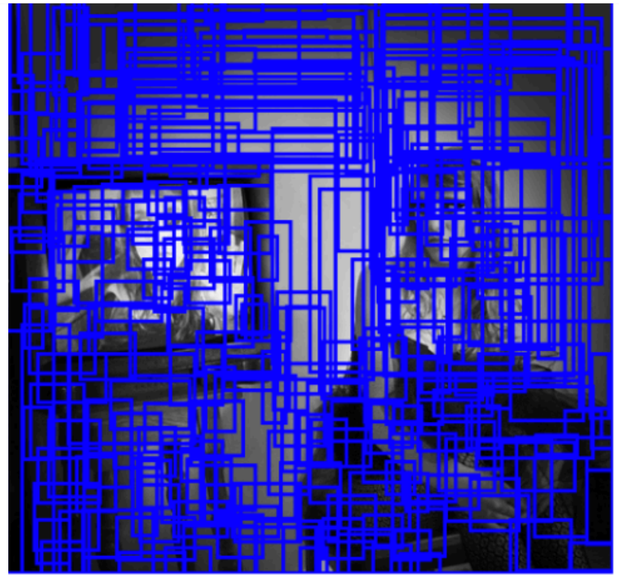
Each bounding box is:
[x,y,w,h]

Searching for objects

Scanning window approach & Image pyramids

Selective search

Input image



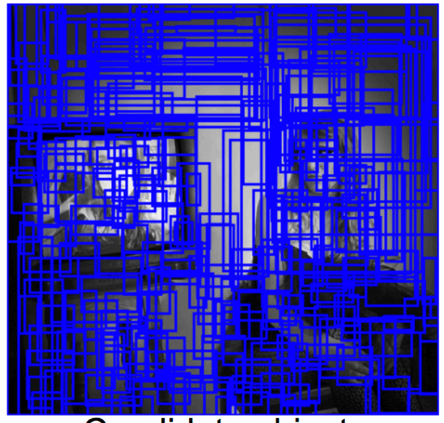
Candidate bounding boxes

Selective search

Stage 1: generate candidate bounding boxes

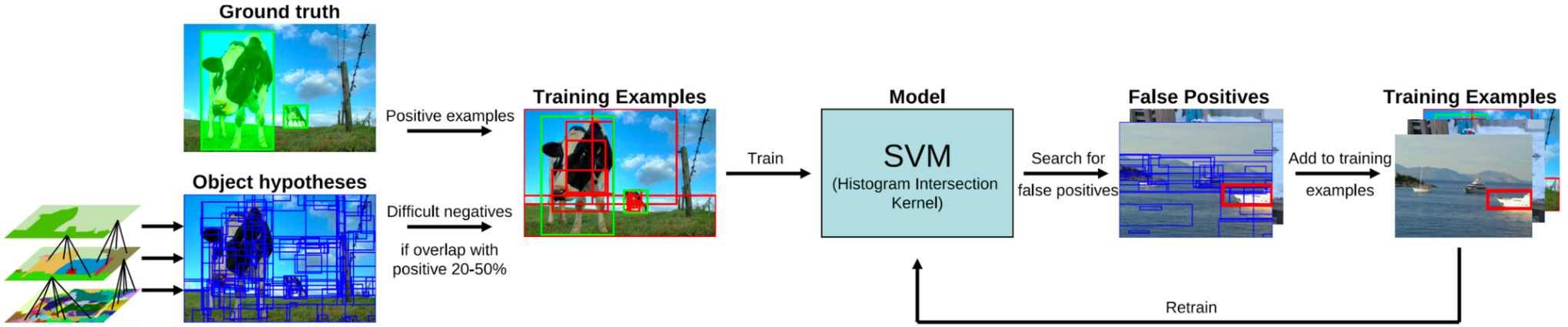
Input Image

Segmentation



Candidate objects

Stage 2: apply classifier to each candidate bounding box



R-CNN, Fast R-CNN, Faster R-CNN

Rich feature hierarchies for accurate object detection and semantic segmentation

Tech report (v5)

Ross Girshick Jeff Donahue Trevor Darrell Jitendra Malik
UC Berkeley
{rbg,jdonahue,trevor,malik}@eecs.berkeley.edu

Abstract

Object detection performance, as measured on the canonical PASCAL VOC dataset, has plateaued in the last few years. The best-performing methods are complex ensemble systems that typically combine multiple low-level image features with high-level context. In this paper, we propose a simple and scalable detection algorithm that improves mean average precision (mAP) by more than 30% relative to the previous best result on VOC 2012—achieving a mAP of 53.3%. Our approach combines two key insights: (1) one can apply high-capacity convolutional neural networks (CNNs) to bottom-up region proposals in order to localize and segment objects and (2) when labeled training data is scarce, supervised pre-training for an auxiliary task, followed by domain-specific fine-tuning, yields a significant performance boost. Since we combine region proposals with CNNs, we call our method R-CNN: *Regions with CNN features. We also compare R-CNN to OverFeat, a recently proposed sliding-window detector based on a similar CNN architecture. We find that R-CNN outperforms OverFeat by a large margin on the 200-class ILSVRC2013 detection dataset. Source code for the complete system is available at* http://www.cs.berkeley.edu/~rbg/rcnn.

1. Introduction

Features matter. The last decade of progress on various visual recognition tasks has been based considerably on the use of SIFT [29] and HOG [7]. But if we look at performance on the canonical visual recognition task, PASCAL VOC object detection [15], it is generally acknowledged that progress has been slow during 2010-2012, with small gains obtained by building ensemble systems and employing minor variants of successful methods.

SIFT and HOG are blockwise orientation histograms, a representation we could associate roughly with complex cells in V1, the first cortical area in the primate visual pathway. But we also know that recognition occurs several stages downstream, which suggests that there might be hierarchical, multi-stage processes for computing features that are even more informative for visual recognition.

Fukushima's "neocognitron" [19], a biologically-inspired hierarchical and shift-invariant model for pattern recognition, was an early attempt at just such a process. The neocognitron, however, lacked a supervised training algorithm. Building on Rumelhart et al. [33], LeCun et al. [26] showed that stochastic gradient descent via back-propagation was effective for training convolutional neural networks (CNNs), a class of models that extend the neocognitron.

CNNs saw heavy use in the 1990s (e.g., [27]), but then fell out of fashion with the rise of support vector machines. In 2012, Krizhevsky et al. [25] rekindled interest in CNNs by showing substantially higher image classification accuracy on the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [9, 10]. Their success resulted from training a large CNN on 1.2 million labeled images, together with a few twists on LeCun's CNN (e.g., $\max(x, 0)$ rectifying non-linearities and "dropout" regularization).

The significance of the ImageNet result was vigorously

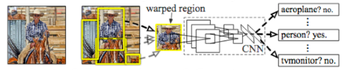


Figure 1: Object detection system overview. Our system (1) takes an input image, (2) extracts around 2000 bottom-up region proposals, (3) computes features for each proposal using a large convolutional neural network (CNN), and then (4) classifies each region using class-specific linear SVMs. R-CNN achieves a mean average precision (mAP) of **53.7% on PASCAL VOC 2010**. For comparison, [39] reports 35.1% mAP using the same region proposals, but with a spatial pyramid and bag-of-visual-words approach. The popular deformable part models perform at 33.4%. On the 200-class **ILSVRC2013 detection dataset, R-CNN's mAP is 31.4%**, a large improvement over OverFeat [34], which had the previous best result at 24.3%.

https://arxiv.org/pdf/1311.2524.pdf

Fast R-CNN

Ross Girshick
Microsoft Research
rbg@microsoft.com

Abstract

This paper proposes a Fast Region-based Convolutional Network method (Fast R-CNN) *for object detection. Fast R-CNN builds on previous work to efficiently classify object proposals using deep convolutional networks. Compared to previous work, Fast R-CNN employs several innovations to improve training and testing speed while also increasing detection accuracy. Fast R-CNN trains the very deep VGG16 network 9× faster than R-CNN, is 213× faster at test-time, and achieves a higher mAP on PASCAL VOC 2012. Compared to SPPnet, Fast R-CNN trains VGG16 3× faster, tests 10× faster, and is more accurate. Fast R-CNN is implemented in Python and C++ (using Caffe) and is available under the open-source MIT License at* https://github.com/rbgirshick/fast-rcnn.

1. Introduction

Recently, deep ConvNets [14, 16] have significantly improved image classification [14] and object detection [9, 19] accuracy. Compared to image classification, object detection is a more challenging task that requires more complex methods to solve. Due to this complexity, current approaches (*e.g.*, [9, 11, 19, 25]) train models in multi-stage pipelines that are slow and inelegant.

Complexity arises because detection requires the accurate localization of objects, creating two primary challenges. First, numerous candidate object locations (often called "proposals") must be processed. Second, these candidates provide only rough localization that must be refined to achieve precise localization. Solutions to these problems often compromise speed, accuracy, or simplicity.

In this paper, we streamline the training process for state-of-the-art ConvNet-based object detectors [9, 11]. We propose a single-stage training algorithm that jointly learns to classify object proposals and refine their spatial locations.

The resulting method can train a very deep detection network (VGG16 [20]) 9× faster than R-CNN [9] and 3× faster than SPPnet [11]. At runtime, the detection network processes images in 0.3s (excluding object proposal time) while achieving top accuracy on PASCAL VOC 2012 [7] with a mAP of 66% (vs. 62% for R-CNN).[1]

1.1. R-CNN and SPPnet

The Region-based Convolutional Network method (R-CNN) [9] achieves excellent object detection accuracy by using a deep ConvNet to classify object proposals. R-CNN, however, has notable drawbacks:

1. **Training is a multi-stage pipeline.** R-CNN first fine-tunes a ConvNet on object proposals using log loss. Then, it fits SVMs to ConvNet features. These SVMs act as object detectors, replacing the softmax classifier learnt by fine-tuning. In the third training stage, bounding-box regressors are learned.
2. **Training is expensive in space and time.** For SVM and bounding-box regressor training, features are extracted from each object proposal in each image and written to disk. With very deep networks, such as VGG16, this process takes 2.5 GPU-days for the 5k images of the VOC07 trainval set. These features require hundreds of gigabytes of storage.
3. **Object detection is slow.** At test-time, features are extracted from each object proposal in each test image. Detection with VGG16 takes 47s / image (on a GPU).

R-CNN is slow because it performs a ConvNet forward pass for each object proposal, without sharing computation. Spatial pyramid pooling networks (SPPnets) [11] were proposed to speed up R-CNN by sharing computation. The SPPnet method computes a convolutional feature map for the entire input image and then classifies each object proposal using a feature vector extracted from the shared feature map. Features are extracted for a proposal by max-pooling the portion of the feature map inside the proposal into a fixed-size output (*e.g.*, 6×6). Multiple output sizes are pooled and then concatenated as in spatial pyramid pooling [15]. SPPnet accelerates R-CNN by 10 to 100× at test time. Training time is also reduced by 3× due to faster proposal feature extraction.

[1] All timings use one Nvidia K40 GPU overclocked to 875 MHz.

https://arxiv.org/pdf/1504.08083.pdf

Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun

Abstract—State-of-the-art object detection networks depend on region proposal algorithms to hypothesize object locations. Advances like SPPnet [1] and Fast R-CNN [2] have reduced the running time of these detection networks, exposing region proposal computation as a bottleneck. In this work, we introduce a *Region Proposal Network* (RPN) that shares full-image convolutional features with the detection network, thus enabling nearly cost-free region proposals. An RPN is a fully convolutional network that simultaneously predicts object bounds and objectness scores at each position. The RPN is trained end-to-end to generate high-quality region proposals, which are used by Fast R-CNN for detection. We further merge RPN and Fast R-CNN into a single network by sharing their convolutional features—using the recently popular terminology of neural networks with "attention" mechanisms, the RPN component tells the unified network where to look. For the very deep VGG-16 model [3], our detection system has a frame rate of 5fps (*including all steps*) on a GPU, while achieving state-of-the-art object detection accuracy on PASCAL VOC 2007, 2012, and MS COCO datasets with only 300 proposals per image. In ILSVRC and COCO 2015 competitions, Faster R-CNN and RPN are the foundations of the 1st-place winning entries in several tracks. Code has been made publicly available.

Index Terms—Object Detection, Region Proposal, Convolutional Neural Network.

1 Introduction

Recent advances in object detection are driven by the success of region proposal methods (*e.g.*, [4]) and region-based convolutional neural networks (R-CNNs) [5]. Although region-based CNNs were computationally expensive as originally developed in [5], their cost has been drastically reduced thanks to sharing convolutions across proposals [1], [2]. The latest incarnation, Fast R-CNN [2], achieves near real-time rates using very deep networks [3], *when ignoring the time spent on region proposals*. Now, proposals are the test-time computational bottleneck in state-of-the-art detection systems.

Region proposal methods typically rely on inexpensive features and economical inference schemes. Selective Search [4], one of the most popular methods, greedily merges superpixels based on engineered low-level features. Yet when compared to efficient detection networks [2], Selective Search is an order of magnitude slower, at 2 seconds per image in a CPU implementation. EdgeBoxes [6] currently provides the best tradeoff between proposal quality and speed, at 0.2 seconds per image. Nevertheless, the region proposal step still consumes as much running time as the detection network.

- *S. Ren is with University of Science and Technology of China, Hefei, China. This work was done when S. Ren was an intern at Microsoft Research. Email: sqren@mail.ustc.edu.cn*
- *K. He and J. Sun are with Visual Computing Group, Microsoft Research. E-mail: {kahe,jiansun}@microsoft.com*
- *R. Girshick is with Facebook AI Research. The majority of this work was done when R. Girshick was with Microsoft Research. E-mail: rbg@fb.com*

One may note that fast region-based CNNs take advantage of GPUs, while the region proposal methods used in research are implemented on the CPU, making such runtime comparisons inequitable. An obvious way to accelerate proposal computation is to re-implement it for the GPU. This may be an effective engineering solution, but re-implementation ignores the down-stream detection network and therefore misses important opportunities for sharing computation.

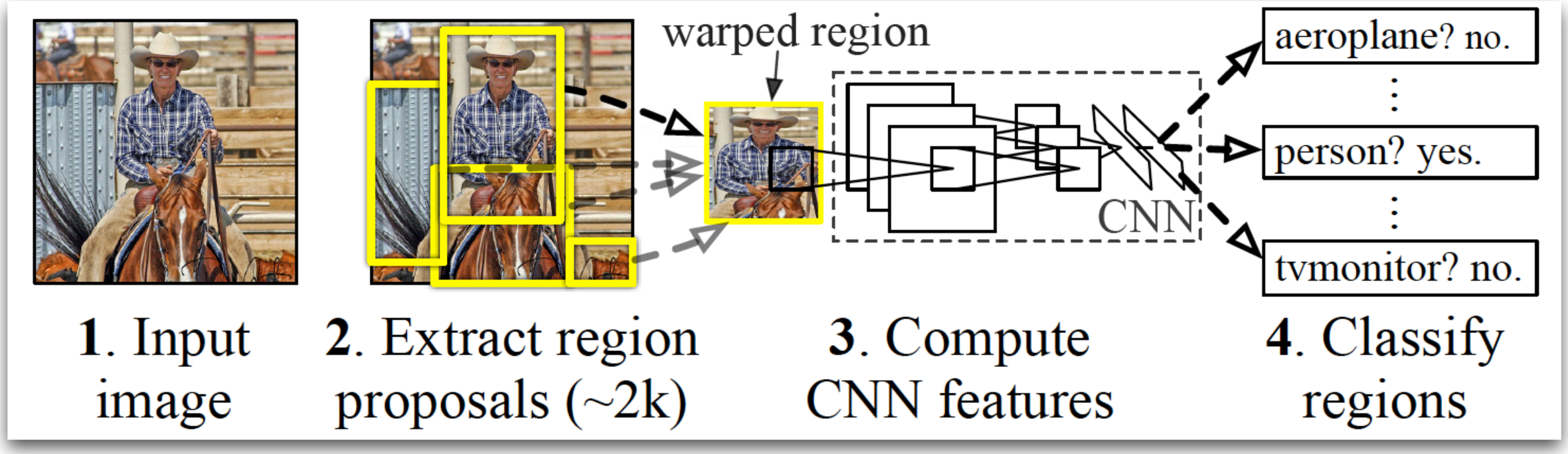
In this paper, we show that an algorithmic change—computing proposals with a deep convolutional neural network—leads to an elegant and effective solution where proposal computation is nearly cost-free given the detection network's computation. To this end, we introduce novel *Region Proposal Networks* (RPNs) that share convolutional layers with state-of-the-art object detection networks [1], [2]. By sharing convolutions at test-time, the marginal cost for computing proposals is small (*e.g.*, 10ms per image).

Our observation is that the convolutional feature maps used by region-based detectors, like Fast R-CNN, can also be used for generating region proposals. On top of these convolutional features, we construct an RPN by adding a few additional convolutional layers that simultaneously regress region bounds and objectness scores at each location on a regular grid. The RPN is thus a kind of fully convolutional network (FCN) [7] and can be trained end-to-end specifically for the task for generating detection proposals.

RPNs are designed to efficiently predict region proposals with a wide range of scales and aspect ratios. In contrast to prevalent methods [8], [9], [1], [2] that use

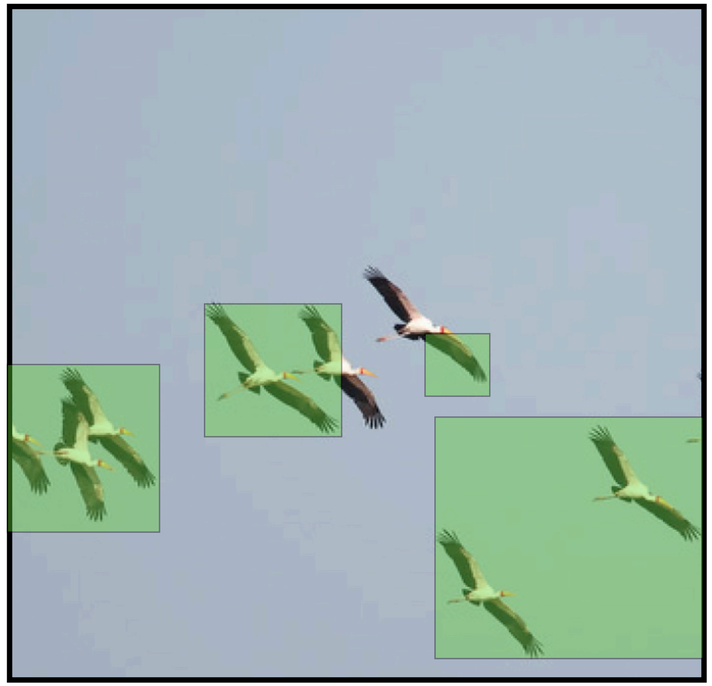
https://arxiv.org/pdf/1506.01497.pdf

R-CNN


warped region
CNN
aeroplane? no.
person? yes.
tvmonitor? no.
1. Input image
2. Extract region proposals (~2k)
3. Compute CNN features
4. Classify regions

R-CNN

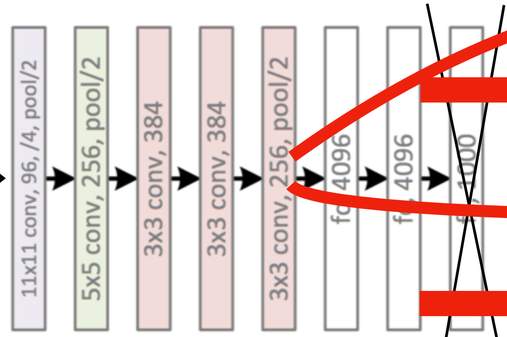
Selective search



~ 2000 region proposals

227×227

AlexNet, ImageNet pretrained
then fine-tuned on the
20 VOC classes



SVM → [x,y,w,h]

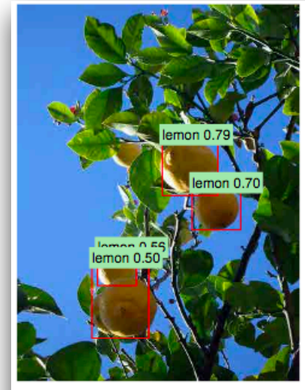
SVM → bird/no bird

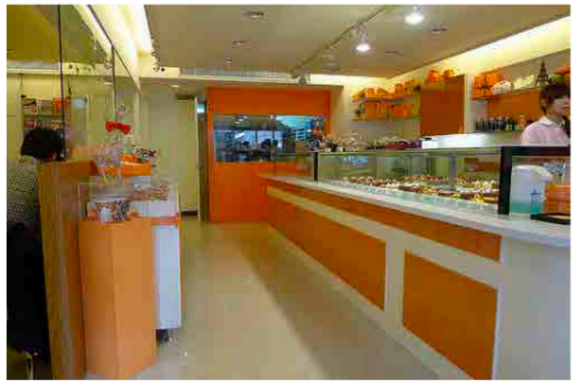
SVM → [x,y,w,h]

SVM → car/no car

⋮

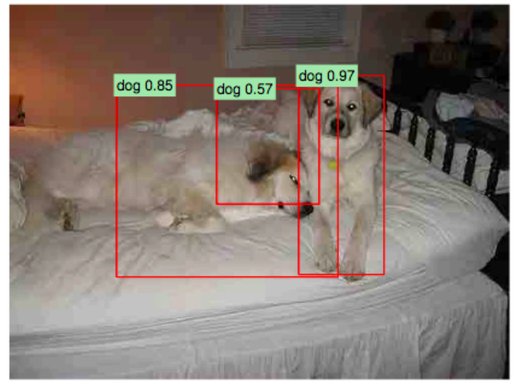
Non-maximal suppression

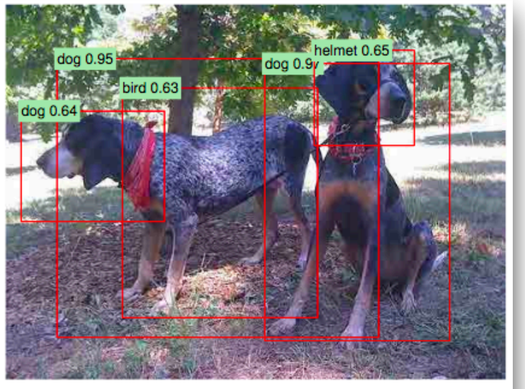

lemon 0.79
lemon 0.70
lemon 0.50

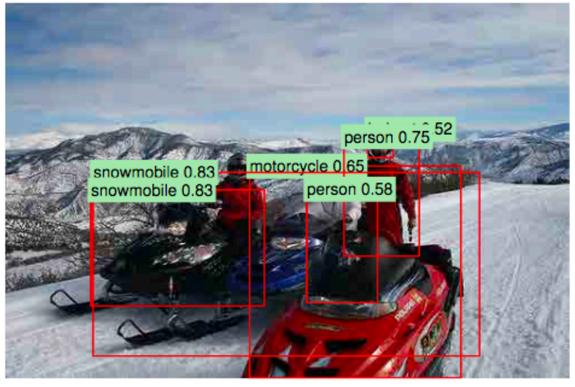


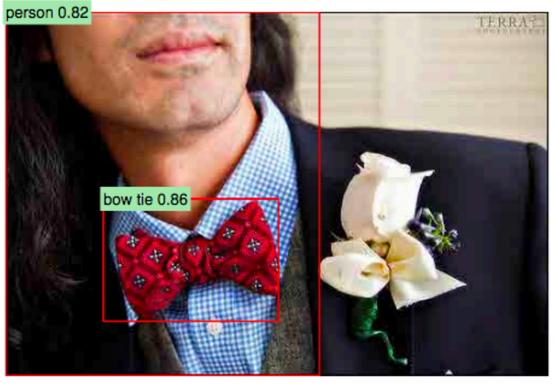

person 0.88
person 0.72


cocktail shaker 0.56

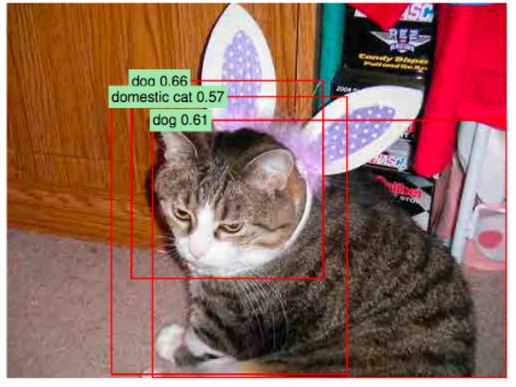

dog 0.85
dog 0.57
dog 0.97

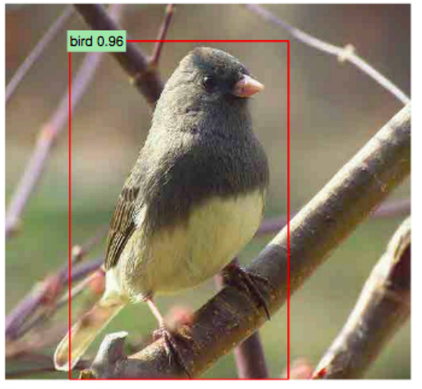

dog 0.95
helmet 0.65
bird 0.63
dog 0.64

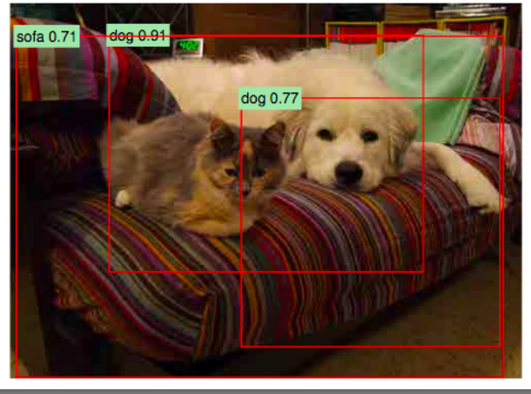

person 0.75
snowmobile 0.83
snowmobile 0.83
person 0.58

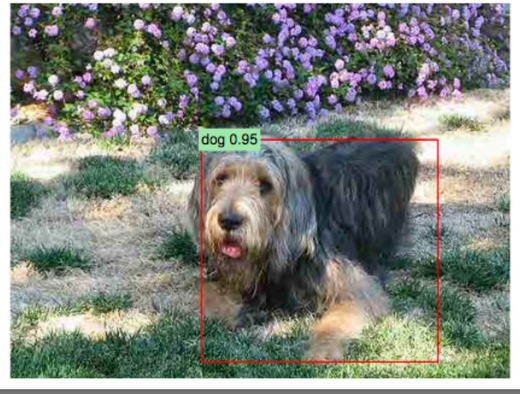

person 0.82
bow tie 0.86

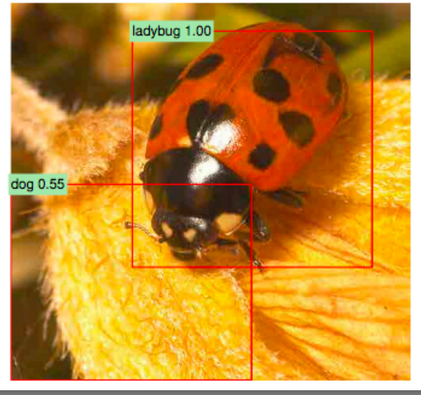

bird 0.61


domestic cat 0.57
dog 0.61


bird 0.96

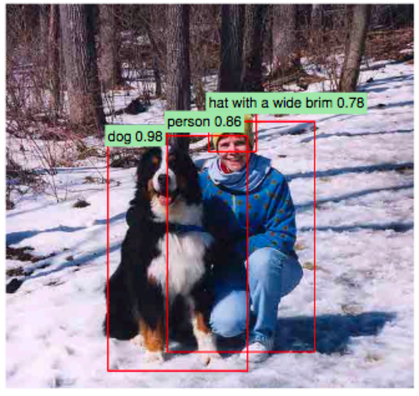

sofa 0.71
dog 0.91
dog 0.77


dog 0.95


ladybug 1.00
dog 0.55


person 0.87


pitcher 0.57


hat with a wide brim 0.78
person 0.86
dog 0.98

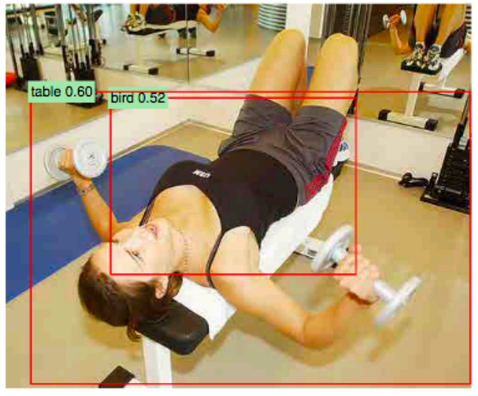
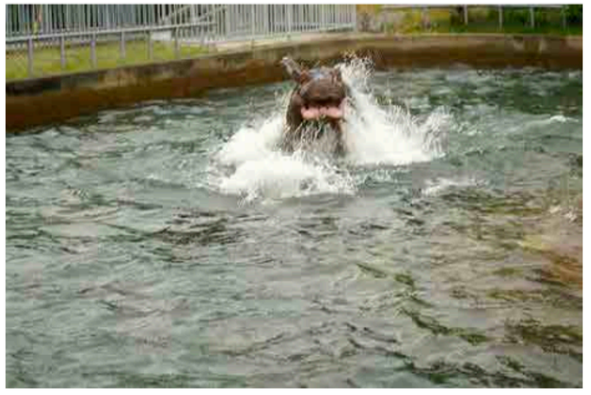


table 0.60
bird 0.52



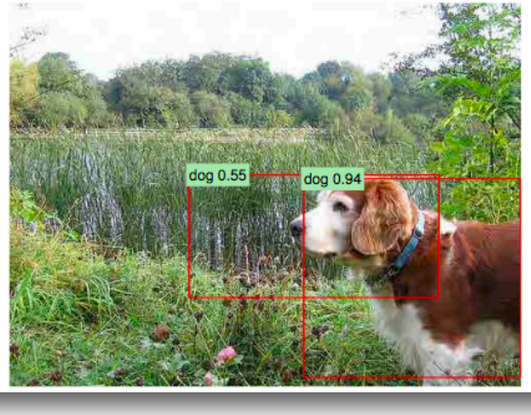

monkey 0.97

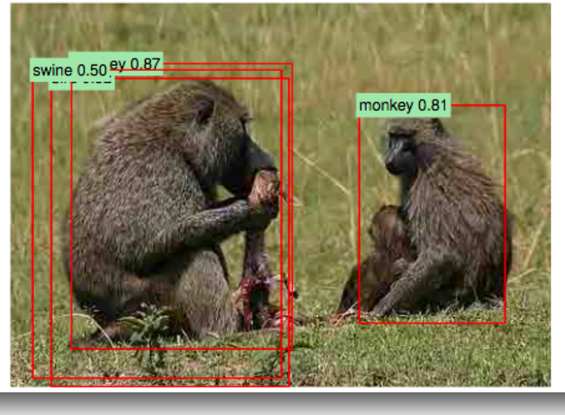

watercraft 0.55
table 0.68


person 0.88


car 0.61

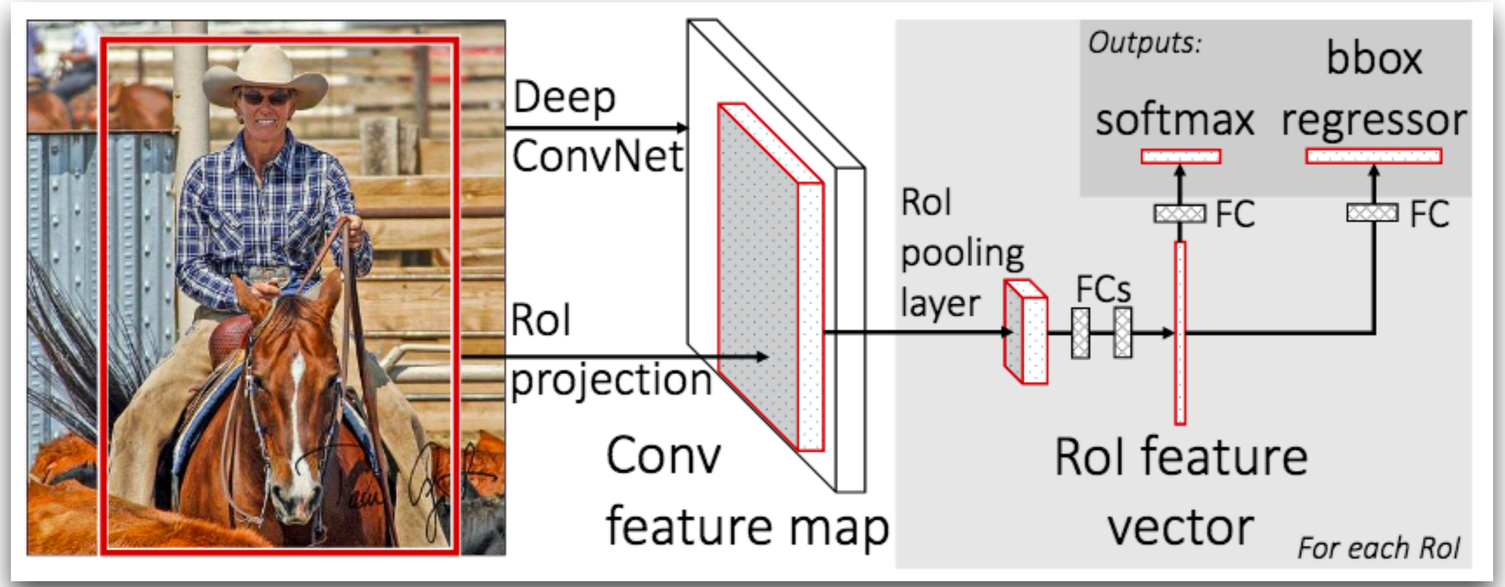

person 0.87
sunglasses 0.51
person 0.51


dog 0.55
dog 0.94


swine 0.50
monkey 0.81

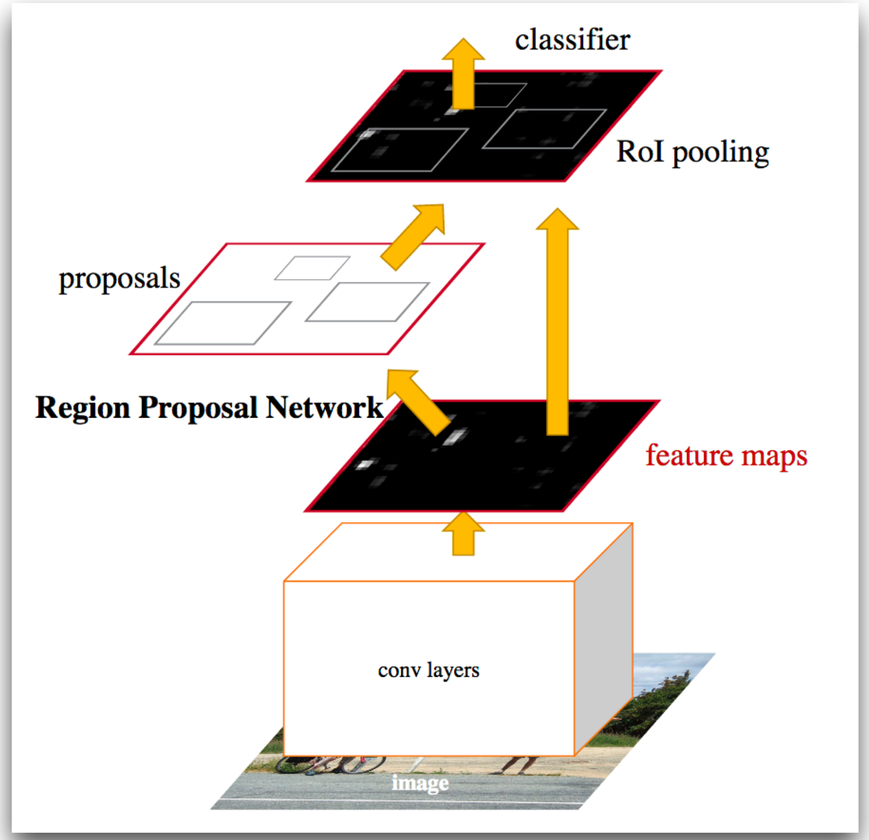

dog 0.97

Making the structure end-to-end



https://arxiv.org/pdf/1504.08083.pdf

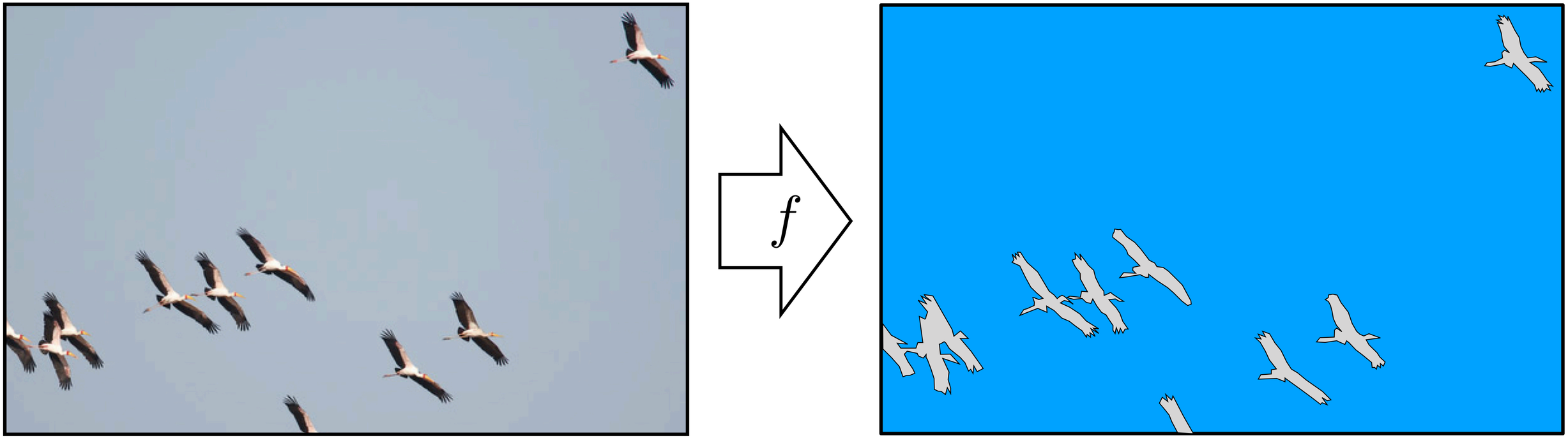
Fast R-CNN



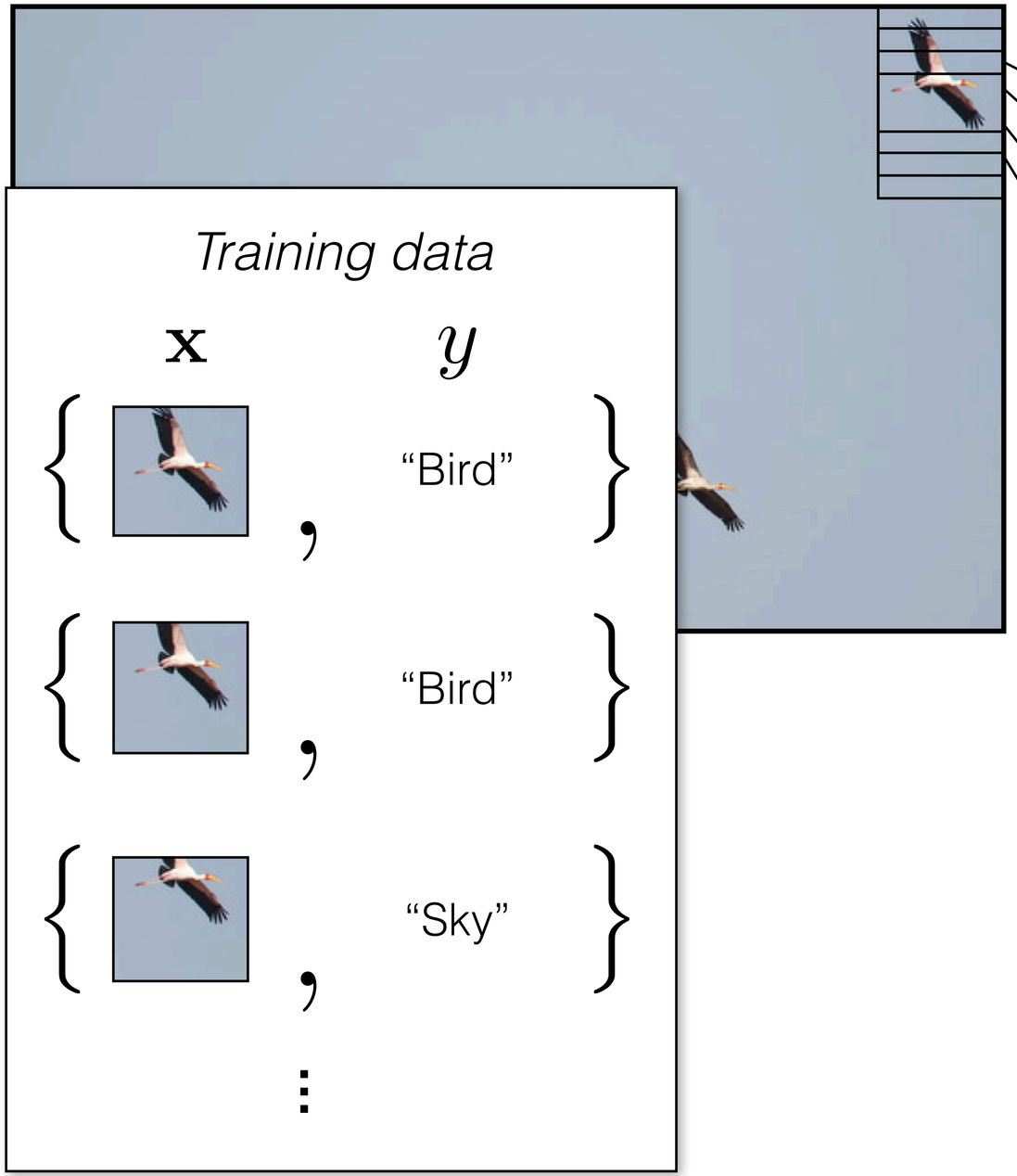
https://arxiv.org/pdf/1506.01497.pdf

Faster R-CNN

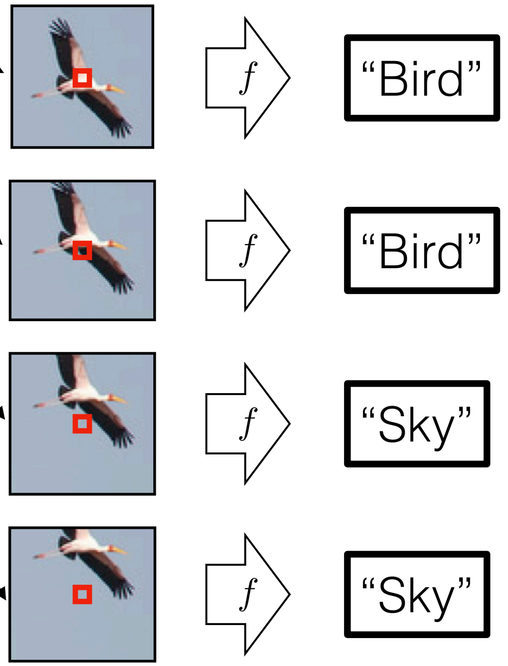
Semantic segmentation



(Colors represent one-hot codes)



What’s the object class of the center pixel?

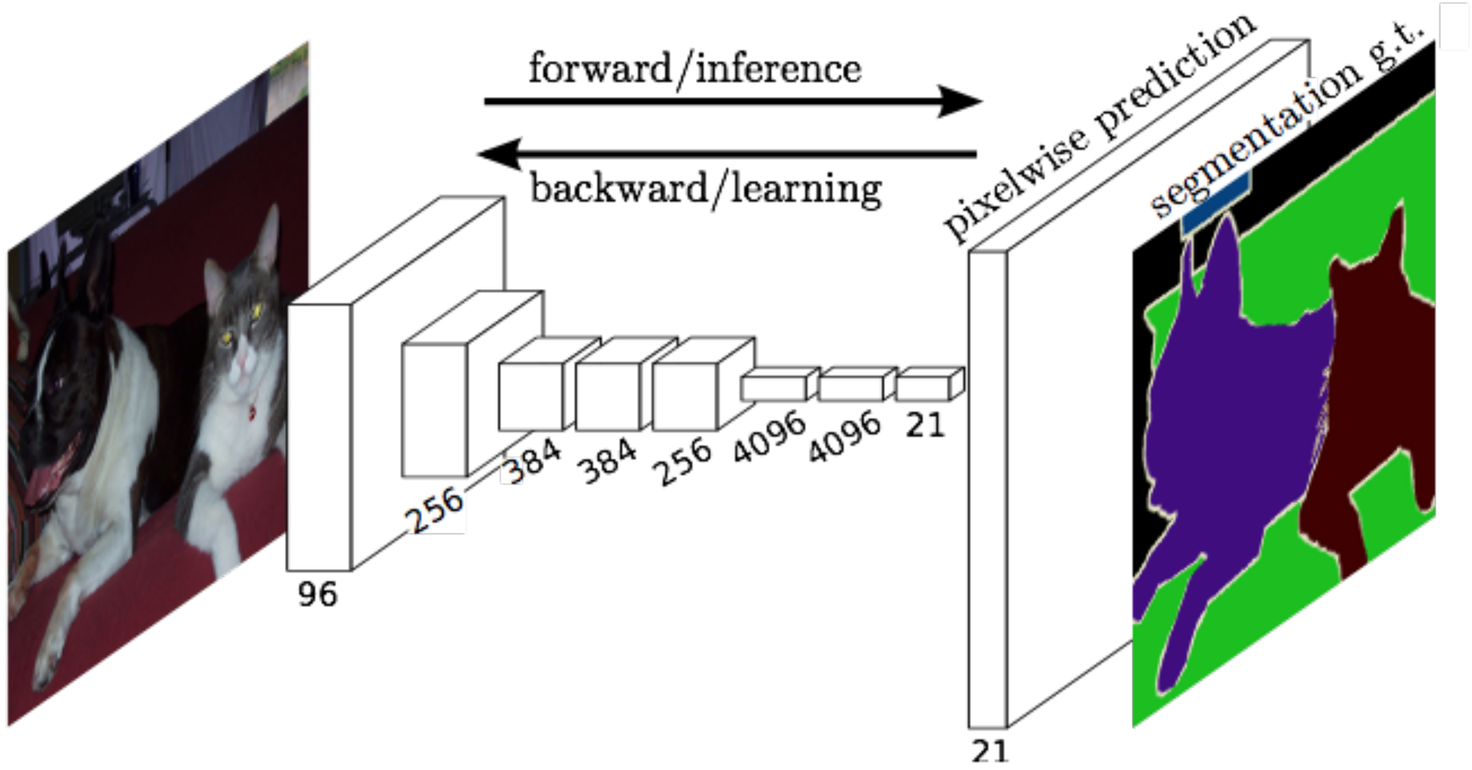


K-way classification problem

Solve with K-dimensional softmax regression:

$$f_\theta : X \to \mathbb{R}^K$$

Fully Convolutional Networks



Fully Convolutional Networks for Semantic Segmentation

Jonathan Long* Evan Shelhamer* Trevor Darrell
UC Berkeley
{jonlong,shelhamer,trevor}@cs.berkeley.edu

Abstract

Convolutional networks are powerful visual models that yield hierarchies of features. We show that convolutional networks by themselves, trained end-to-end, pixels-to-pixels, exceed the state-of-the-art in semantic segmentation. Our key insight is to build "fully convolutional" networks that take input of arbitrary size and produce correspondingly-sized output with efficient inference and learning. We define and detail the space of fully convolutional networks, explain their application to spatially dense prediction tasks, and draw connections to prior models. We adapt contemporary classification networks (AlexNet [22], the VGG net [34], and GoogLeNet [35]) into fully convolutional networks and transfer their learned representations by fine-tuning [5] to the segmentation task. We then define a skip architecture that combines semantic information from a deep, coarse layer with appearance information from a shallow, fine layer to produce accurate and detailed segmentations. Our fully convolutional network achieves state-of-the-art segmentation of PASCAL VOC (20% relative improvement to 62.2% mean IU on 2012), NYUDv2, and SIFT Flow, while inference takes less than one fifth of a second for a typical image.

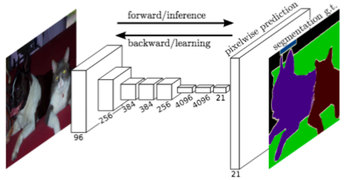


Figure 1. Fully convolutional networks can efficiently learn to make dense predictions for per-pixel tasks like semantic segmentation.

1. Introduction

Convolutional networks are driving advances in recognition. Convnets are not only improving for whole-image classification [22, 34, 35], but also making progress on local tasks with structured output. These include advances in bounding box object detection [32, 12, 19], part and keypoint prediction [42, 26], and local correspondence [26, 10].

The natural next step in the progression from coarse to fine inference is to make a prediction at every pixel. Prior approaches have used convnets for semantic segmentation [30, 3, 9, 31, 17, 15, 11], in which each pixel is labeled with the class of its enclosing object or region, but with shortcomings that this work addresses.

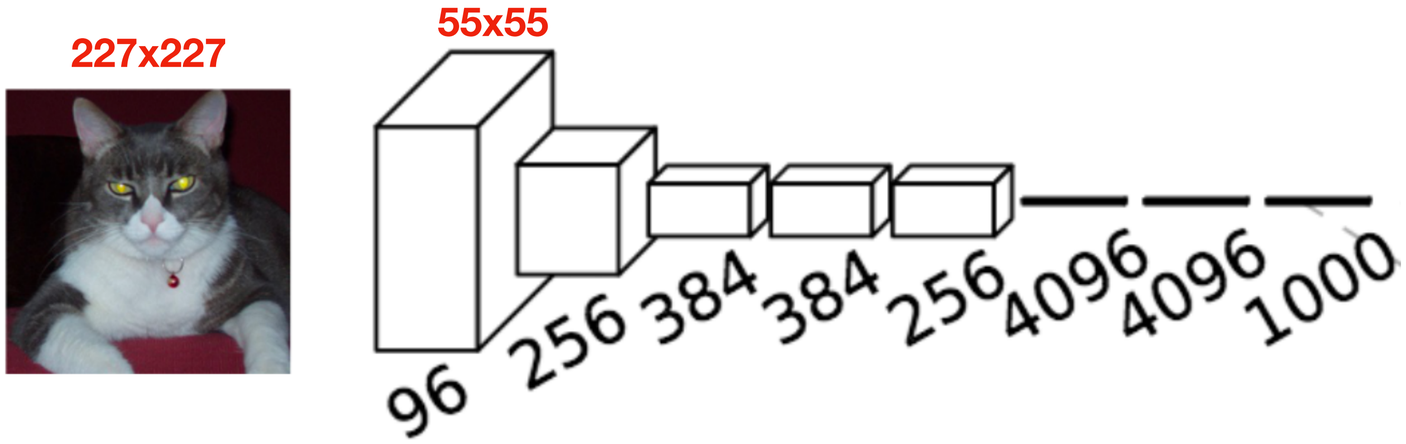
*Authors contributed equally

We show that a fully convolutional network (FCN) trained end-to-end, pixels-to-pixels on semantic segmentation exceeds the state-of-the-art without further machinery. To our knowledge, this is the first work to train FCNs end-to-end (1) for pixelwise prediction and (2) from supervised pre-training. Fully convolutional versions of existing networks predict dense outputs from arbitrary-sized inputs. Both learning and inference are performed whole-image-at-a-time by dense feedforward computation and backpropagation. In-network upsampling layers enable pixelwise prediction and learning in nets with subsampled pooling.

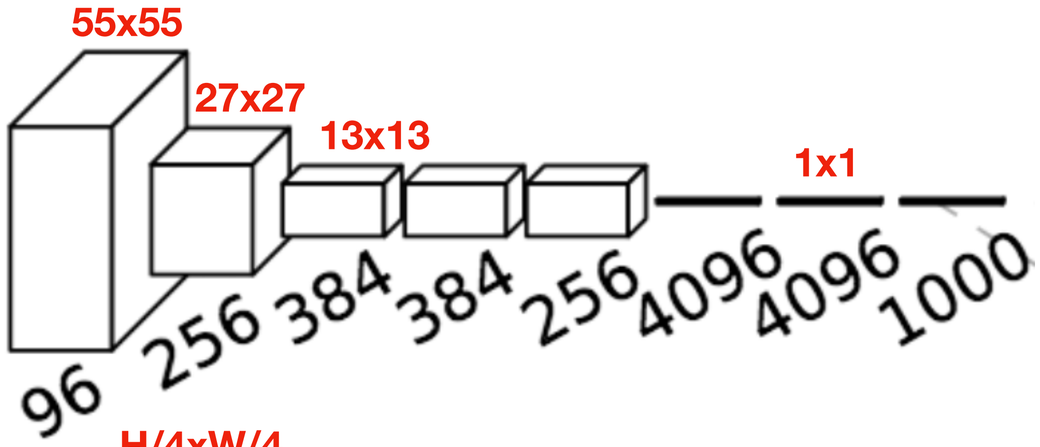
This method is efficient, both asymptotically and absolutely, and precludes the need for the complications in other works. Patchwise training is common [30, 3, 9, 31, 11], but lacks the efficiency of fully convolutional training. Our approach does not make use of pre- and post-processing complications, including superpixels [9, 17], proposals [17, 15], or post-hoc refinement by random fields or local classifiers [9, 17]. Our model transfers recent success in classification [22, 34, 35] to dense prediction by reinterpreting classification nets as fully convolutional and fine-tuning from their learned representations. In contrast, previous works have applied small convnets without supervised pre-training [9, 31, 30].

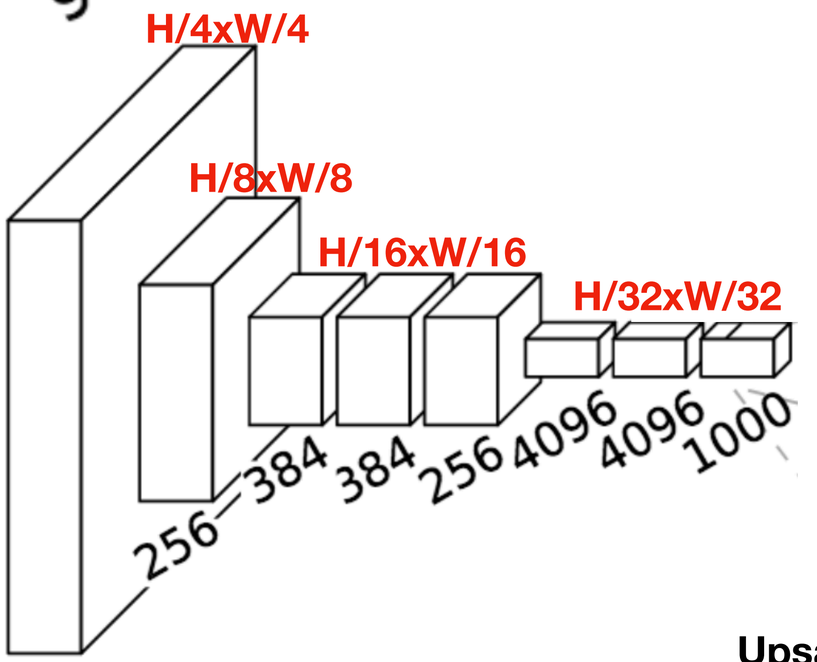
Semantic segmentation faces an inherent tension between semantics and location: global information resolves what while local information resolves where. Deep feature hierarchies encode location and semantics in a nonlinear

Fully Convolutional Networks



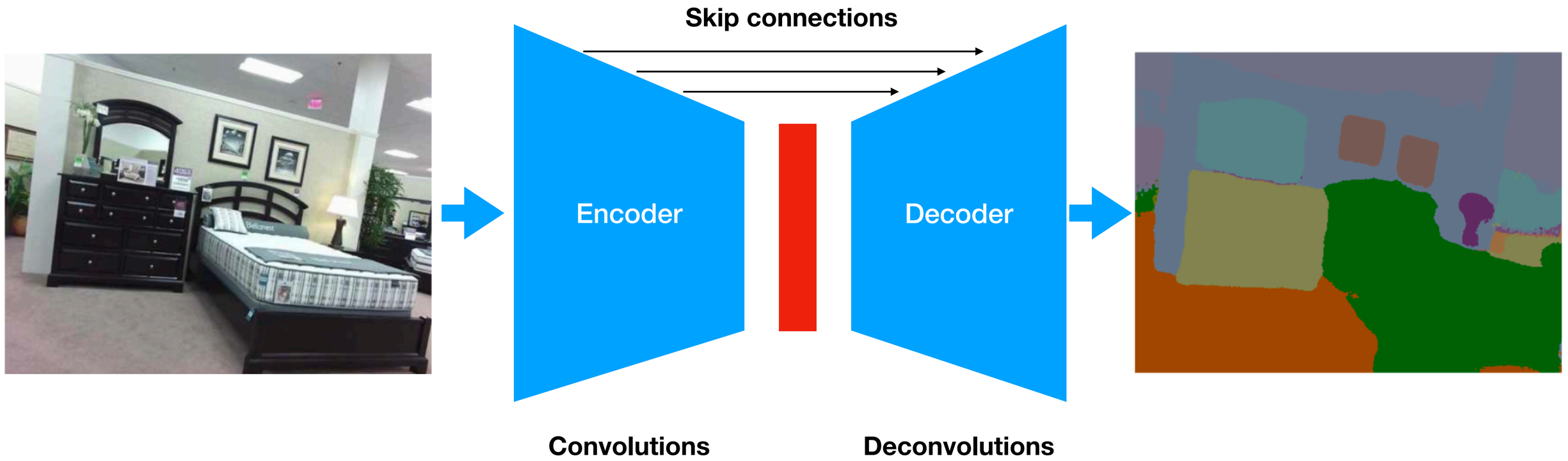
Fully Convolutional Networks





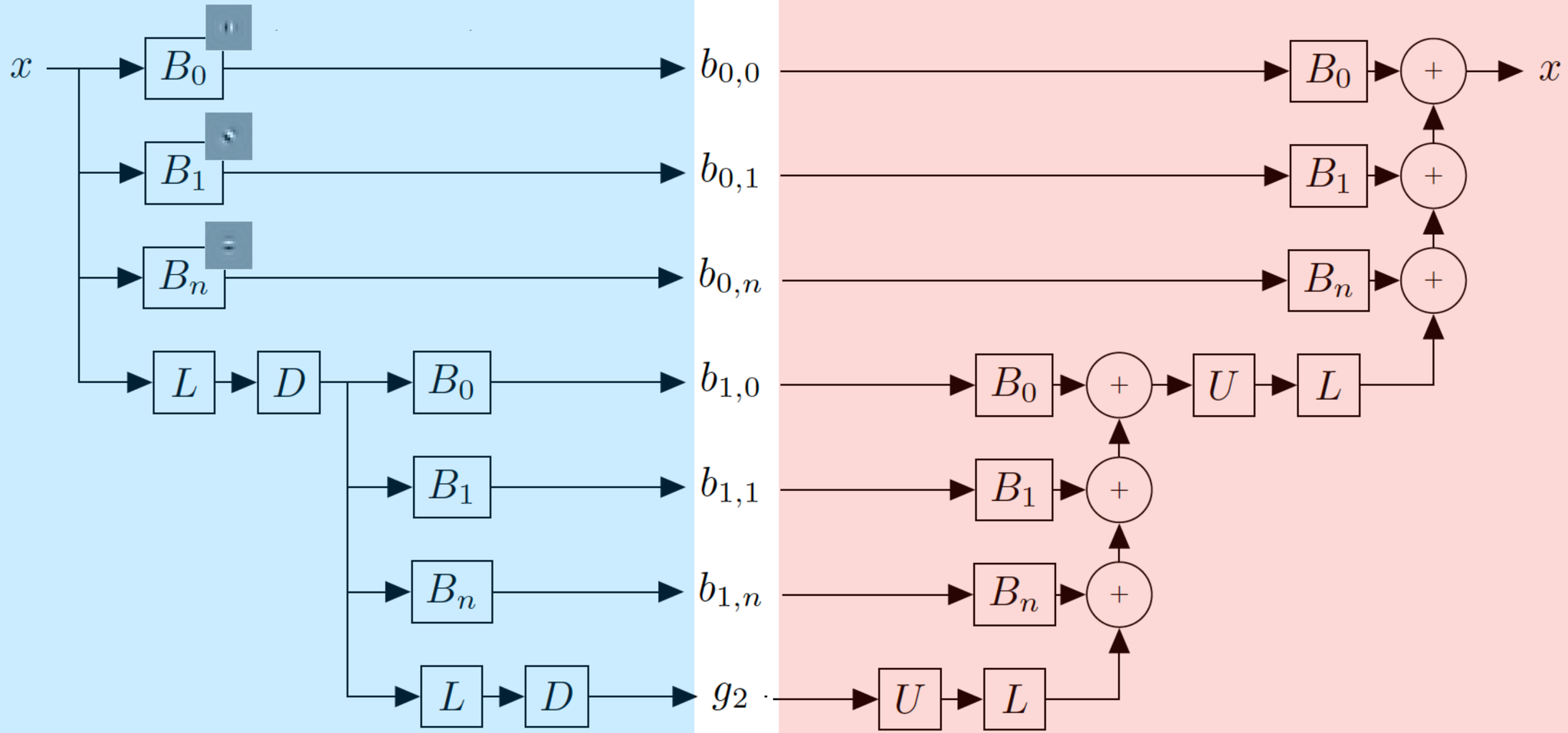
Upsampling

Encoder-decoder architectures



U-Net

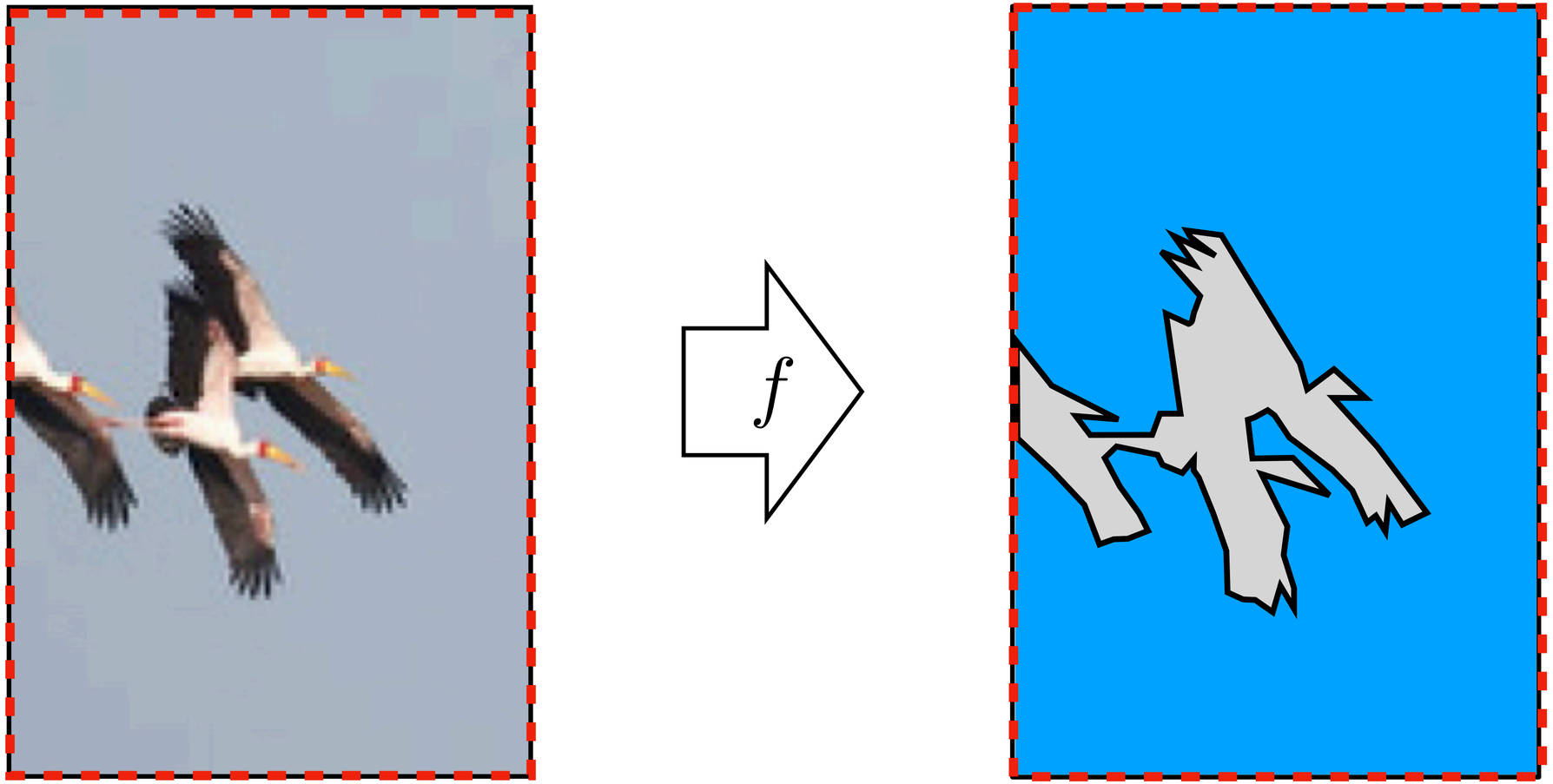
Steerable Pyramid — *A U-Net architecture*



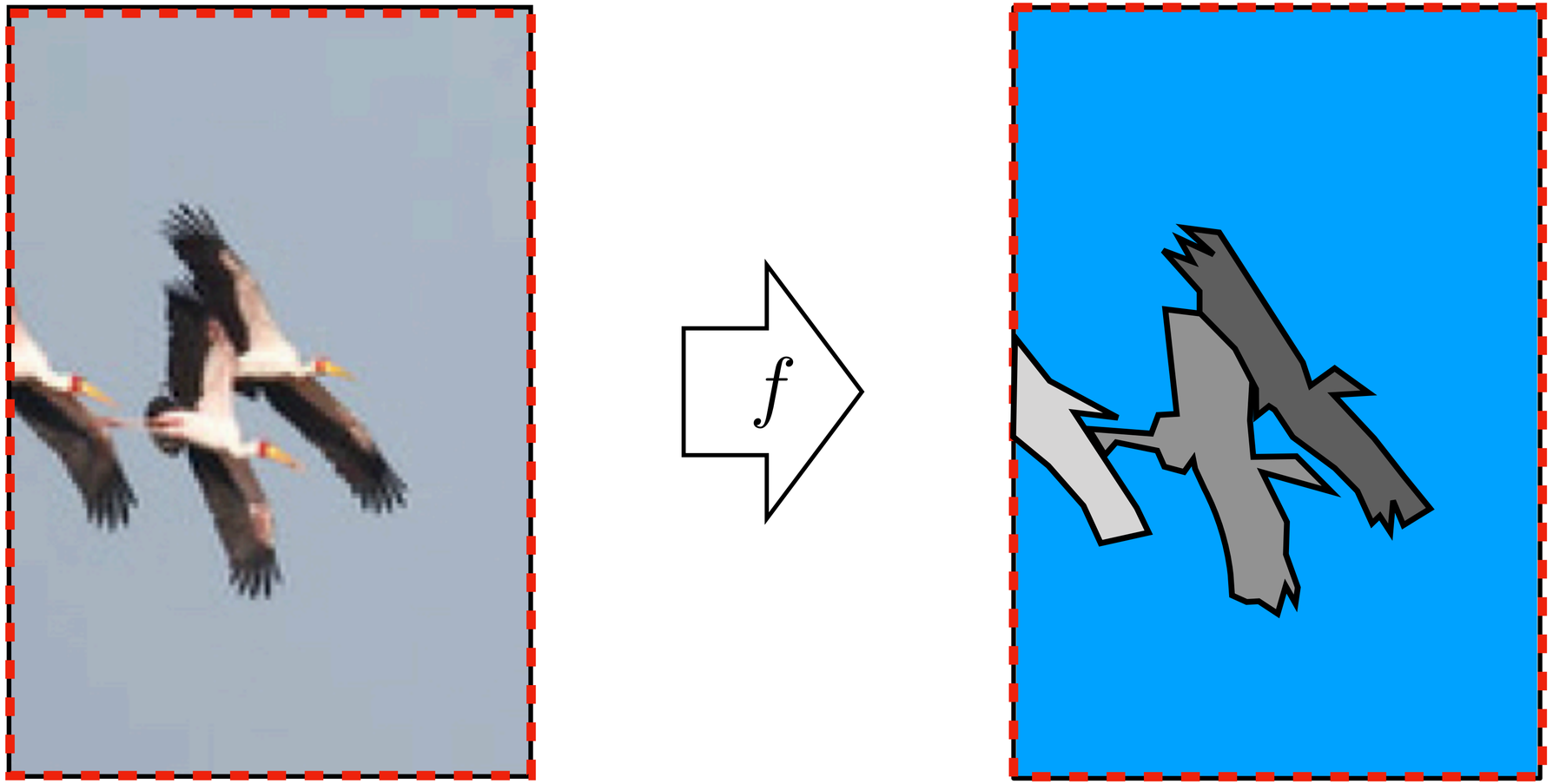
Analysis/Encoder

Synthesis/Decoder

Semantic segmentation



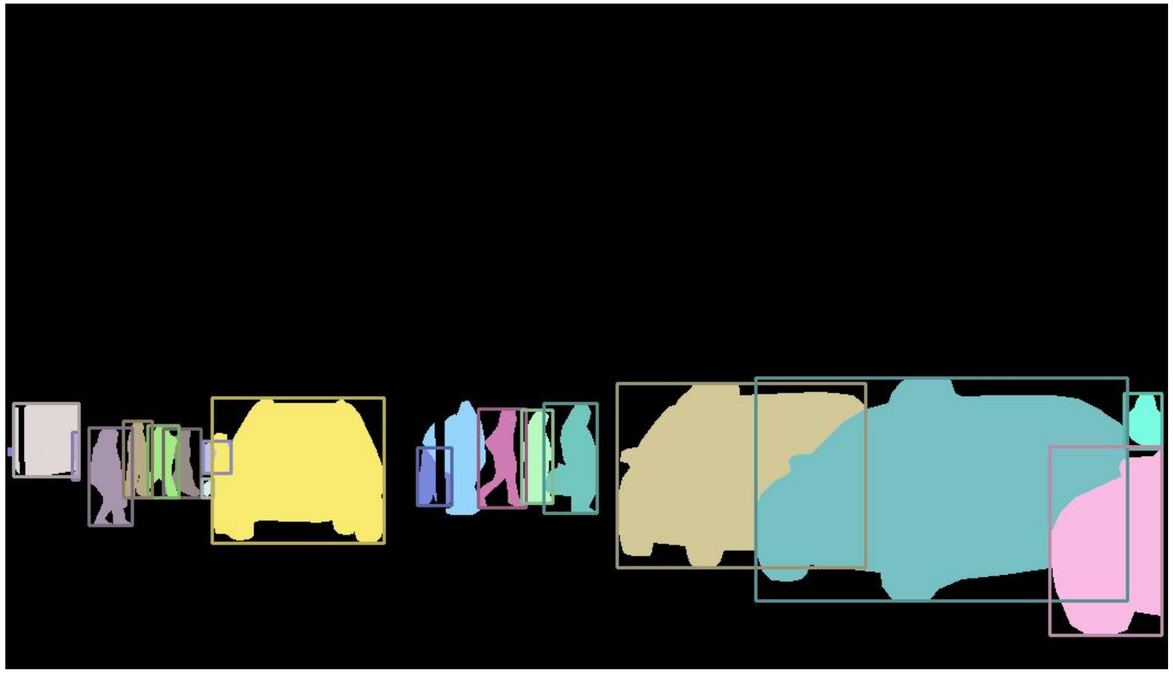
Instance segmentation



Challenge: unbounded number of output instances

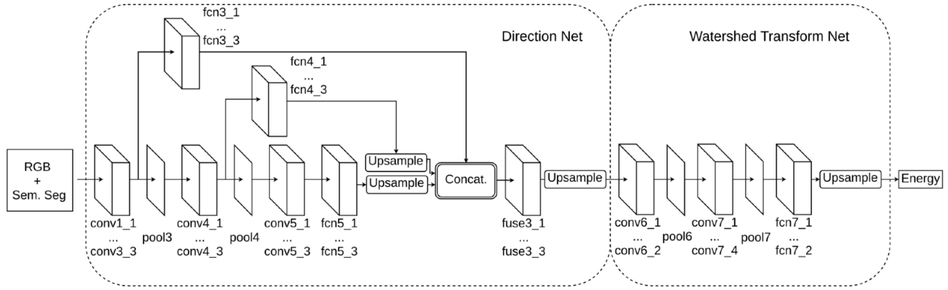
(can’t just do K-way classification)

Instance segmentation

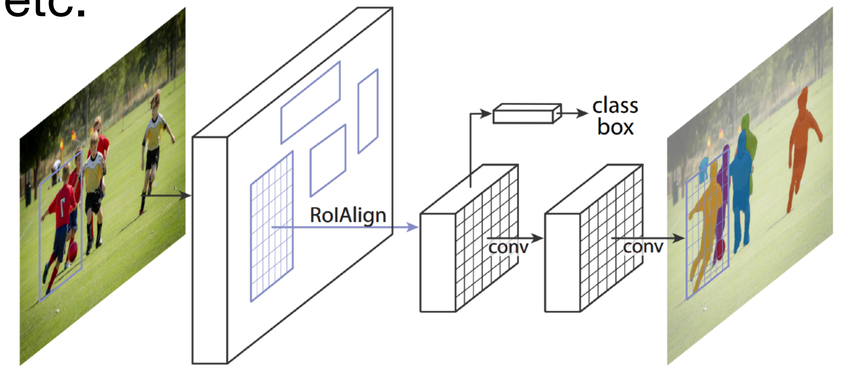


Approaches

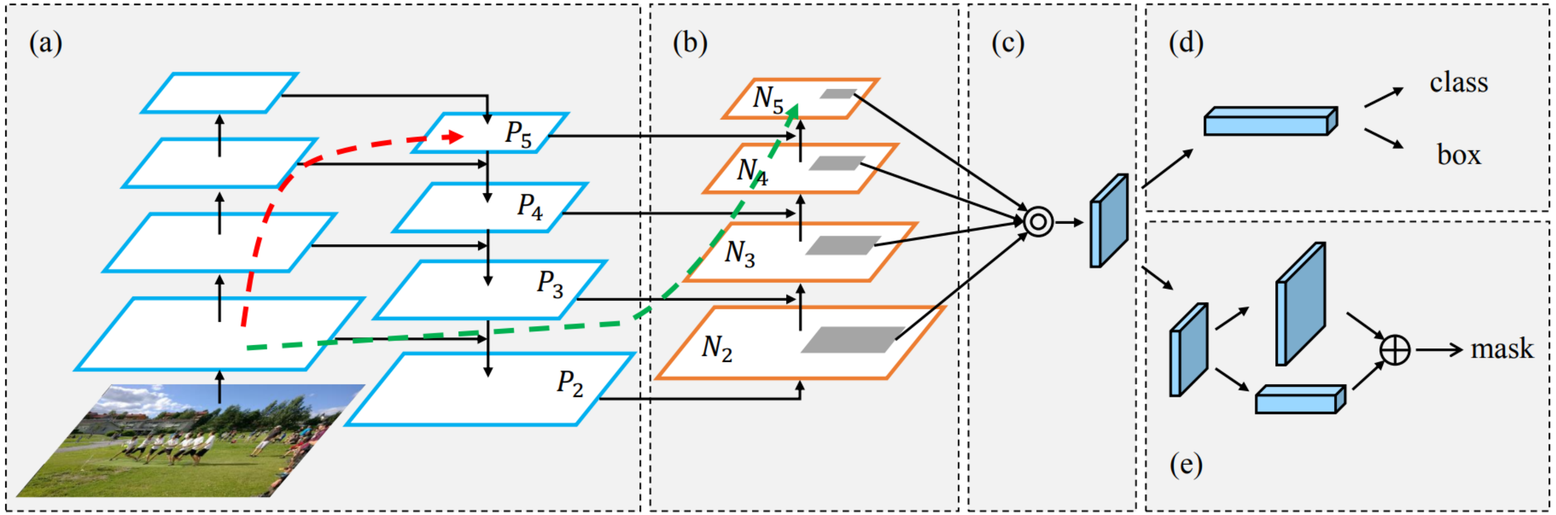
InstanceCut, DWT, SAIS, DIN, FCIS, SGN, Mask-RCNN, PANet etc.



DWT [Bai et al, CVPR'17]

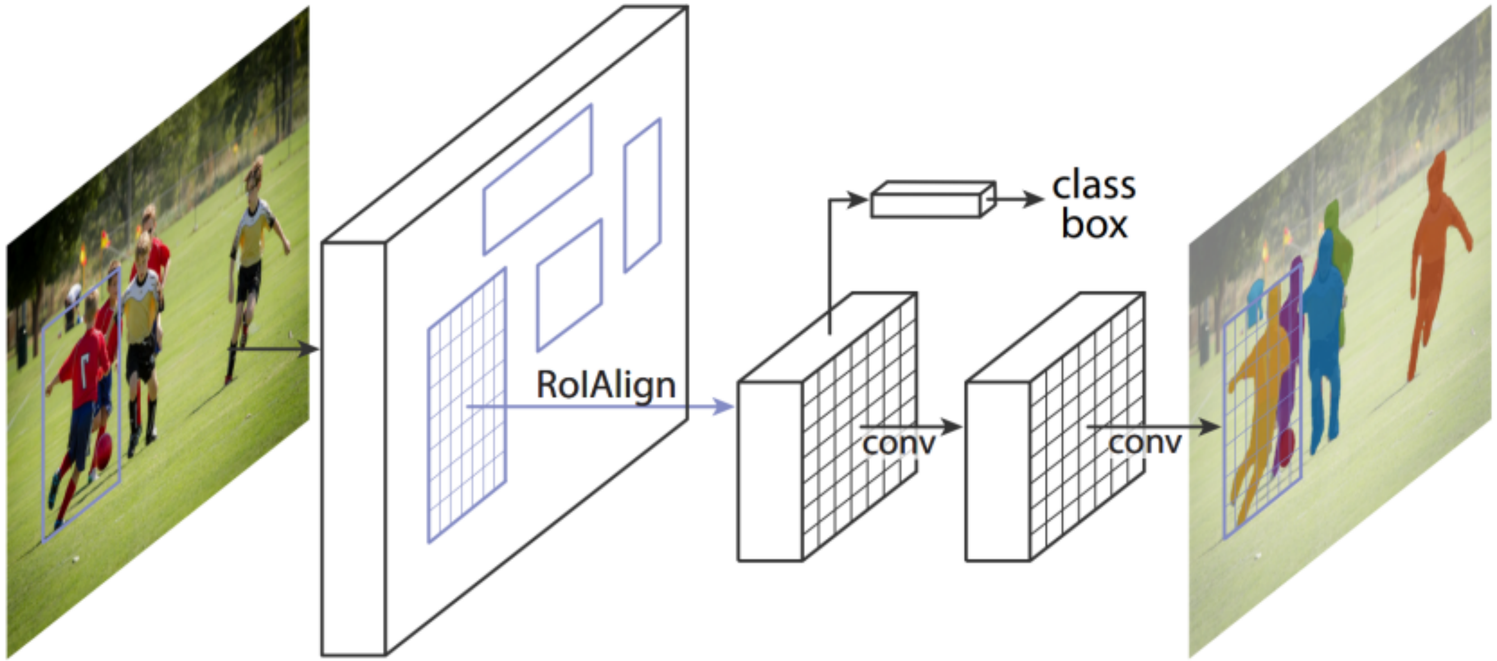


Mask-RCNN [He et al, ICCV'17]



PANet [Liu et al, CVPR'18]

Mask R-CNN



Mask R-CNN

Kaiming He Georgia Gkioxari Piotr Dollár Ross Girshick

Facebook AI Research (FAIR)

Abstract

We present a conceptually simple, flexible, and general framework for object instance segmentation. Our approach efficiently detects objects in an image while simultaneously generating a high-quality segmentation mask for each instance. The method, called Mask R-CNN, extends Faster R-CNN by adding a branch for predicting an object mask in parallel *with the existing branch for bounding box recognition. Mask R-CNN is simple to train and adds only a small overhead to Faster R-CNN, running at 5 fps. Moreover, Mask R-CNN is easy to generalize to other tasks,* e.g., *allowing us to estimate human poses in the same framework. We show top results in all three tracks of the COCO suite of challenges, including instance segmentation, bounding-box object detection, and person keypoint detection. Without bells and whistles, Mask R-CNN outperforms all existing, single-model entries on every task, including the COCO 2016 challenge winners. We hope our simple and effective approach will serve as a solid baseline and help ease future research in instance-level recognition. Code has been made available at:* https://github.com/facebookresearch/Detectron.

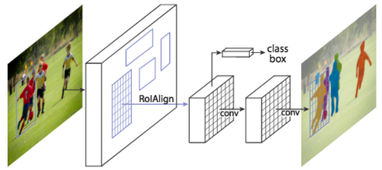


Figure 1. The **Mask R-CNN** framework for instance segmentation.

1. Introduction

The vision community has rapidly improved object detection and semantic segmentation results over a short period of time. In large part, these advances have been driven by powerful baseline systems, such as the Fast/Faster R-CNN [12, 36] and Fully Convolutional Network (FCN) [30] frameworks for object detection and semantic segmentation, respectively. These methods are conceptually intuitive and offer flexibility and robustness, together with fast training and inference time. Our goal in this work is to develop a comparably enabling framework for *instance segmentation*.

Instance segmentation is challenging because it requires the correct detection of all objects in an image while also precisely segmenting each instance. It therefore combines elements from the classical computer vision tasks of *object detection*, where the goal is to classify individual objects and localize each using a bounding box, and *semantic segmentation*, where the goal is to classify each pixel into a fixed set of categories without differentiating object instances.[1] Given this, one might expect a complex method is required to achieve good results. However, we show that a surprisingly simple, flexible, and fast system can surpass prior state-of-the-art instance segmentation results.

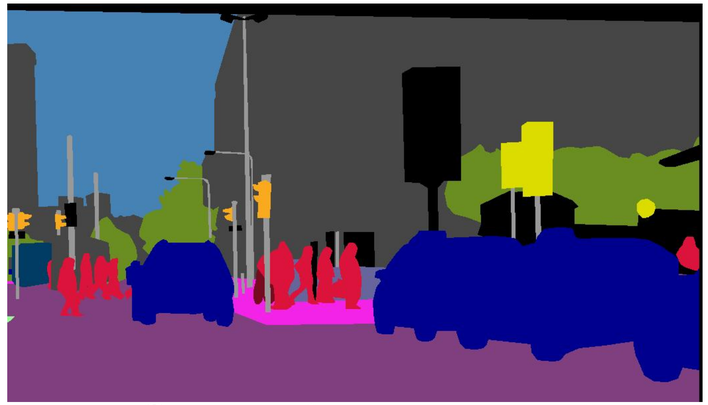
Our method, called *Mask R-CNN*, extends Faster R-CNN [36] by adding a branch for predicting segmentation masks on each Region of Interest (RoI), in *parallel* with the existing branch for classification and bounding box regression (Figure 1). The mask branch is a small FCN applied to each RoI, predicting a segmentation mask in a pixel-to-pixel manner. Mask R-CNN is simple to implement and train given the Faster R-CNN framework, which facilitates a wide range of flexible architecture designs. Additionally, the mask branch only adds a small computational overhead, enabling a fast system and rapid experimentation.

In principle Mask R-CNN is an intuitive extension of Faster R-CNN, yet constructing the mask branch properly is critical for good results. Most importantly, Faster R-CNN was not designed for pixel-to-pixel alignment between network inputs and outputs. This is most evident in how *RoIPool* [18, 12], the *de facto* core operation for attending to instances, performs coarse spatial quantization for feature extraction. To fix the misalignment, we propose a simple, quantization-free layer, called *RoIAlign*, that faithfully preserves exact spatial locations. Despite being

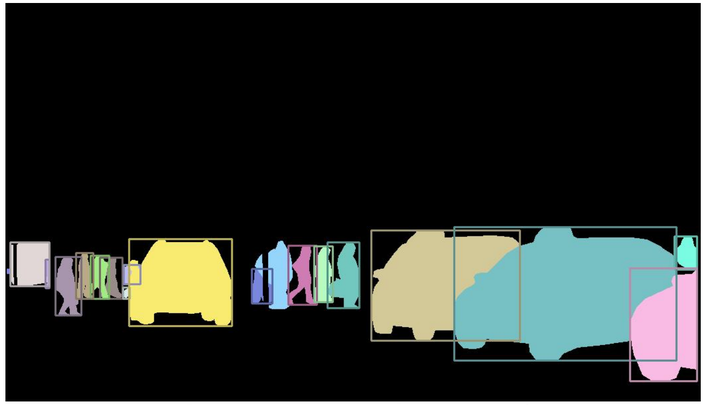
[1] Following common terminology, we use *object detection* to denote detection via *bounding boxes*, not masks, and *semantic segmentation* to denote per-pixel classification without differentiating instances. Yet we note that *instance segmentation* is both semantic and a form of detection.

https://arxiv.org/abs/1703.06870

Panoptic Segmentation



Semantic segmentation

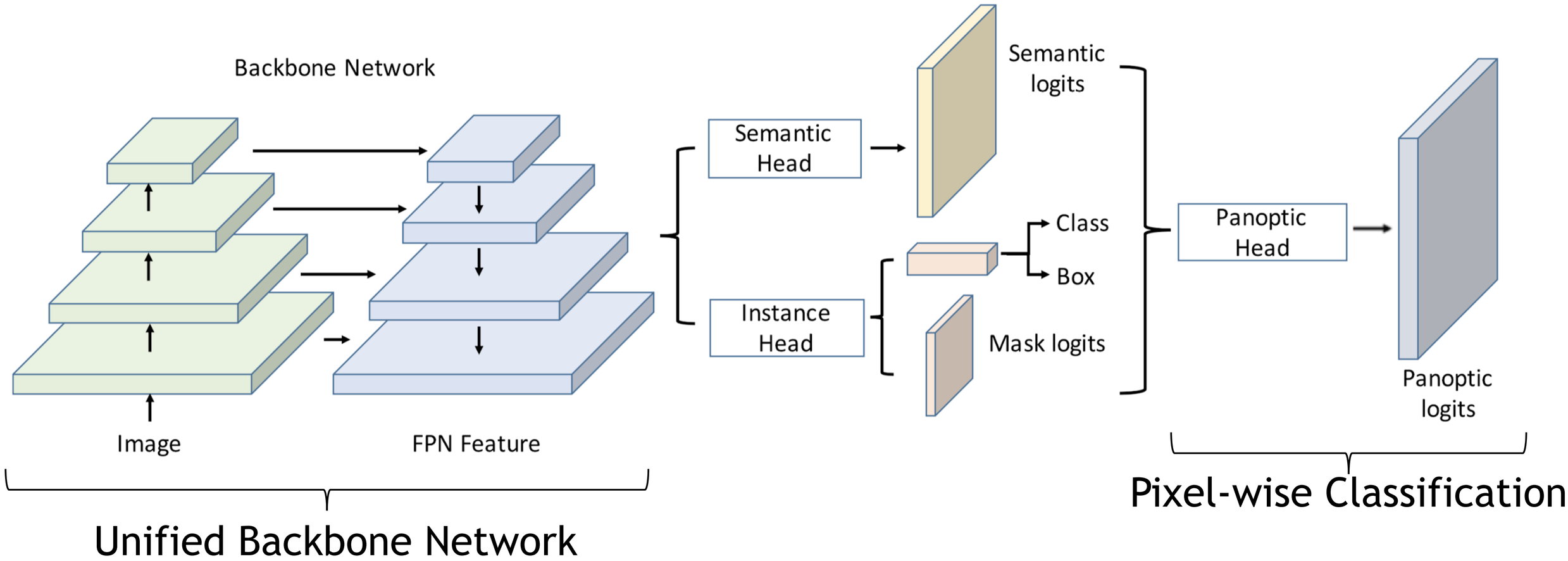


Instance detection

panoptic segmentation:

stuff and things are solved, instances distinguishable

Unified Panoptic Segmentation Network (UPSNet)




UPSNet-101-M: Cityscapes

Depth Perception

a	0	0
0	a	0
0	0	1

.

R

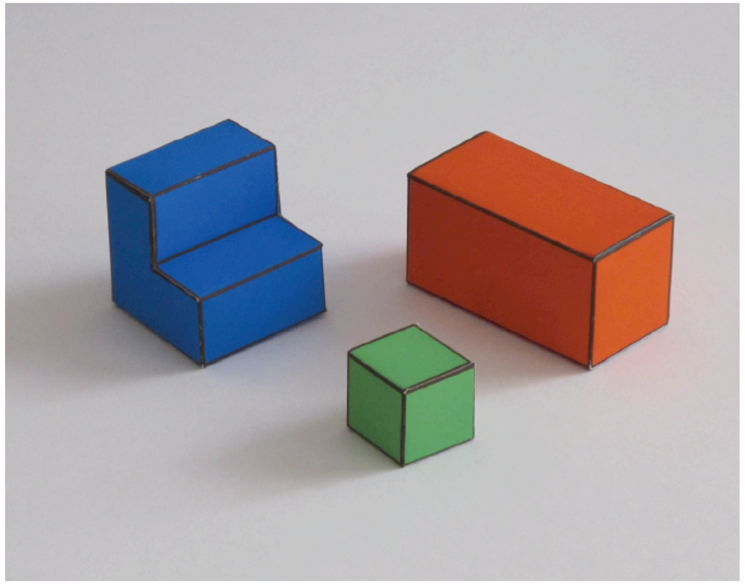
.

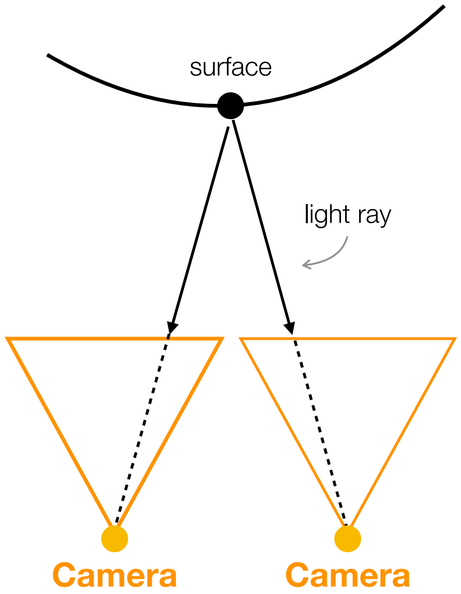
I	-T

x
y
w

X_w
Y_w
Z_w
1

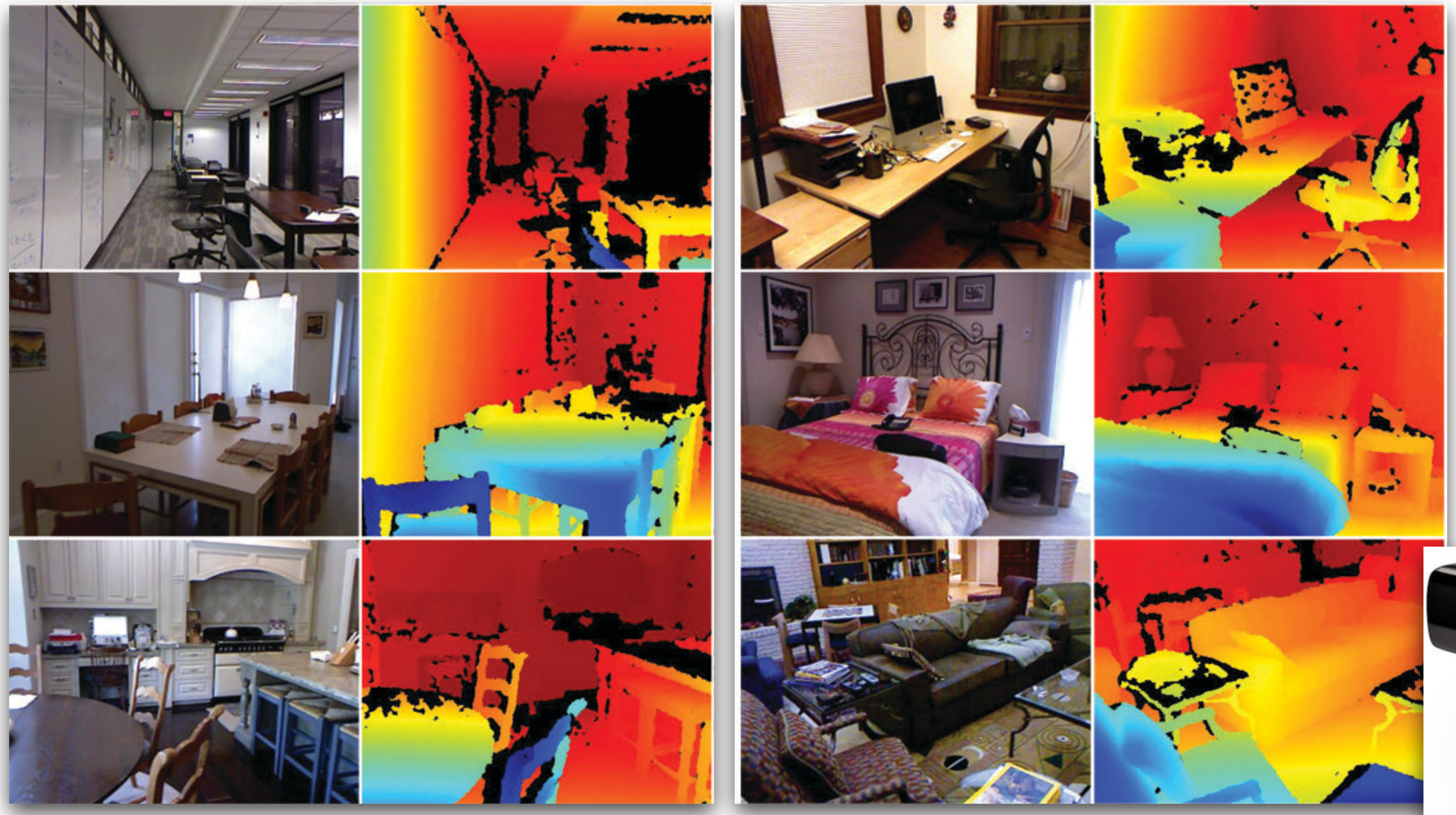
?





3D scene understanding in the deep net era

3D in the deep learning era

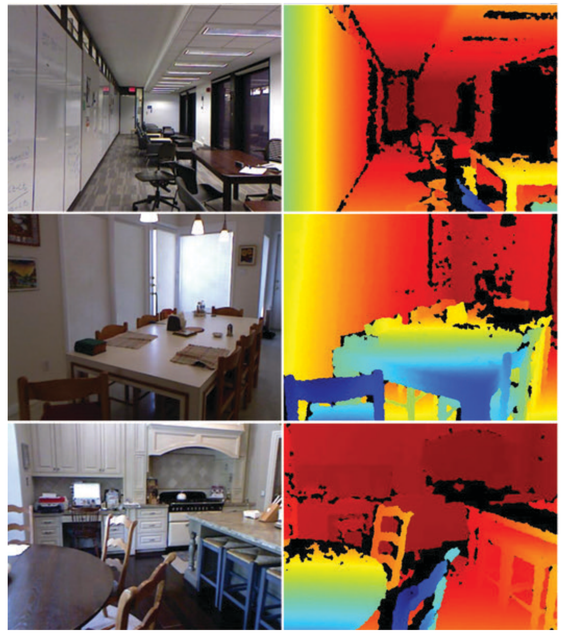


Ground truth is collected by using traditional methods

3D in the deep learning era

Data

$\{\mathbf{x}^{(i)}, \mathbf{y}^{(i)}\}_{i=1}^{N}$



$\rightarrow$

Learner

Objective

scale invariant MSE in log space

Hypothesis space

Deep Neural Network

Optimizer

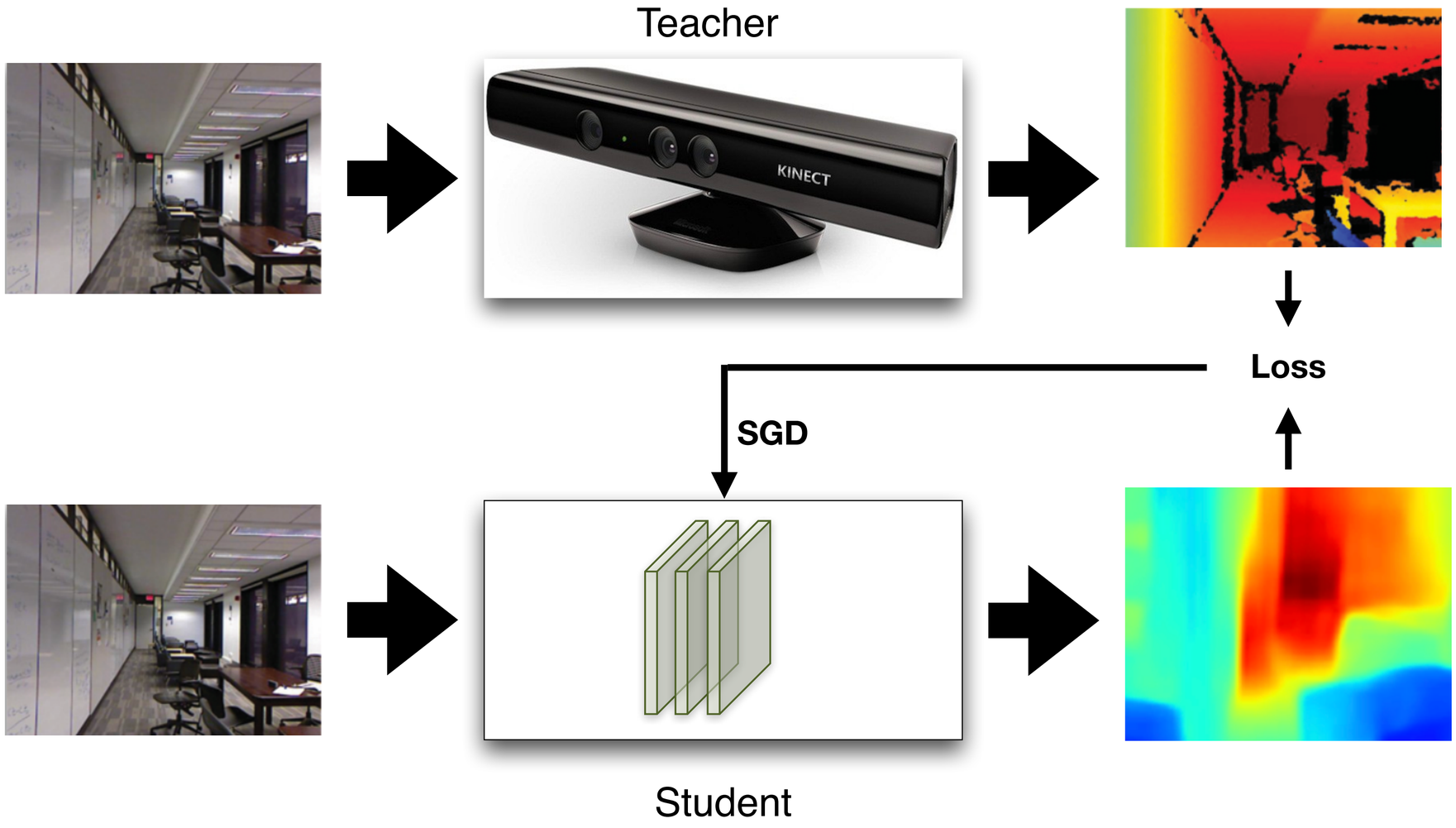
SGD

$\rightarrow f$

Regular old supervised learning!

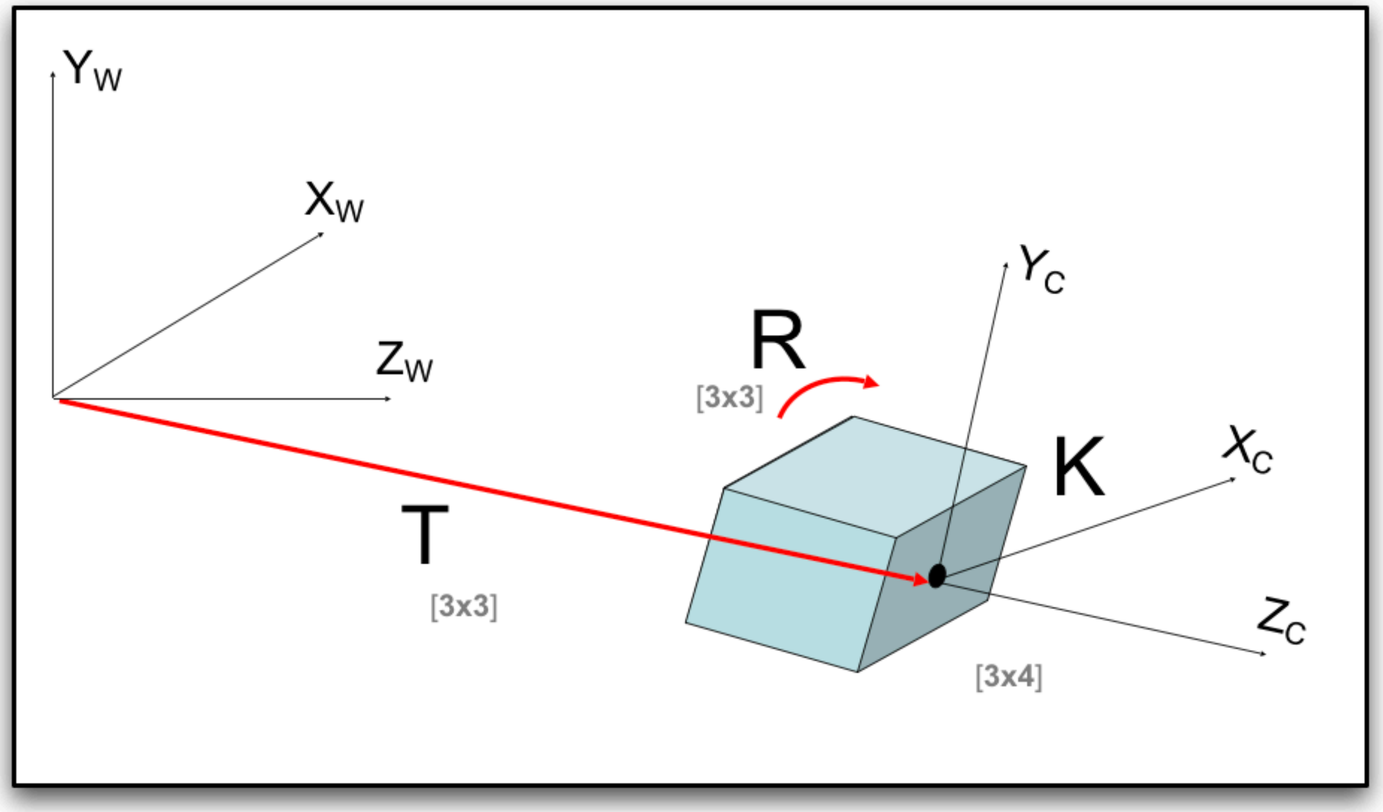
$$f^* = \arg\min_{f \in \mathcal{F}} \mathcal{L}(f(\mathbf{x}^{(i)}), \mathbf{y}^{(i)})$$

3D in the deep learning era



Scale invariant error

With uncalibrated cameras (unknown K), the global scale of a scene is an “ambiguity” in depth prediction.



… you could learn estimate K from a single image …

Scale invariant error

Estimate log depth instead of depth. Defining y_i as the ground truth depth on pixel i, and y^*_i its estimated depth:

Standard L2 error:

$$D_{L2}(y, y^*) = \frac{1}{n}\sum_{i=1}^{n}(\log y_i - \log y_i^*)^2$$

Scale invariant error:

$$D_{SI}(y, y^*) = \frac{1}{n}\sum_{i=1}^{n}(\log y_i - \log y_i^* + \alpha(y, y^*))^2$$

with

$$\alpha(y, y^*) = \frac{1}{n}\sum_{j=1}^{n}(\log y_j - \log y_j^*)$$

[Eigen & Fergus, NIPS, 2014]

Training:

- **Training loss**: Mixture of both error measures (best \lambda=0.5):

Standard L2 error: **Scale invariant error:**

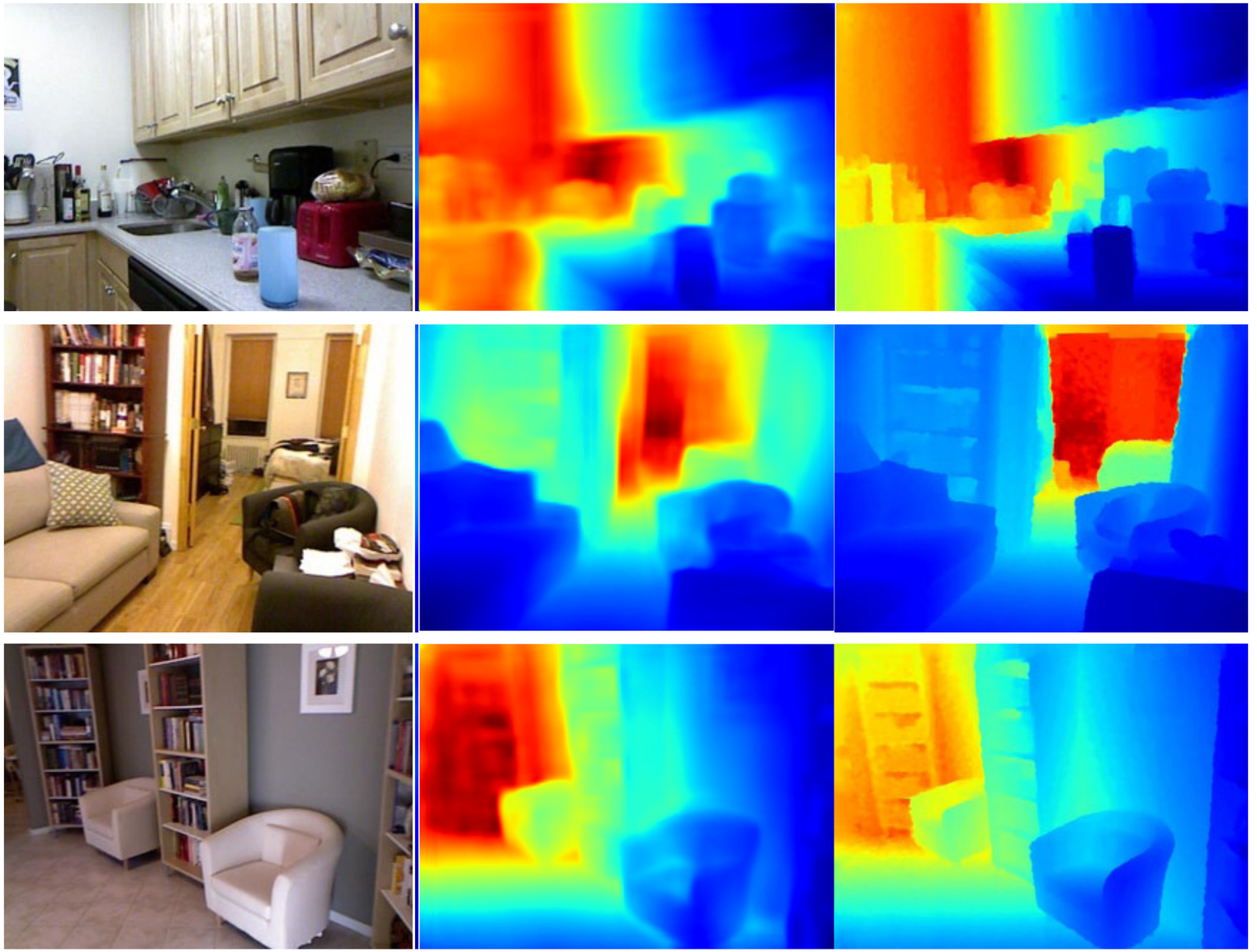
$$J = \lambda D_{L2}(y, y^*) + (1 - \lambda) D_{SI}(y, y^*)$$

Depth contains missing values. Only evaluate on valid pixels.

- **Data augmentation:** flips, translations, scalings, color scalings, …

[Eigen & Fergus, NIPS, 2014]

Results



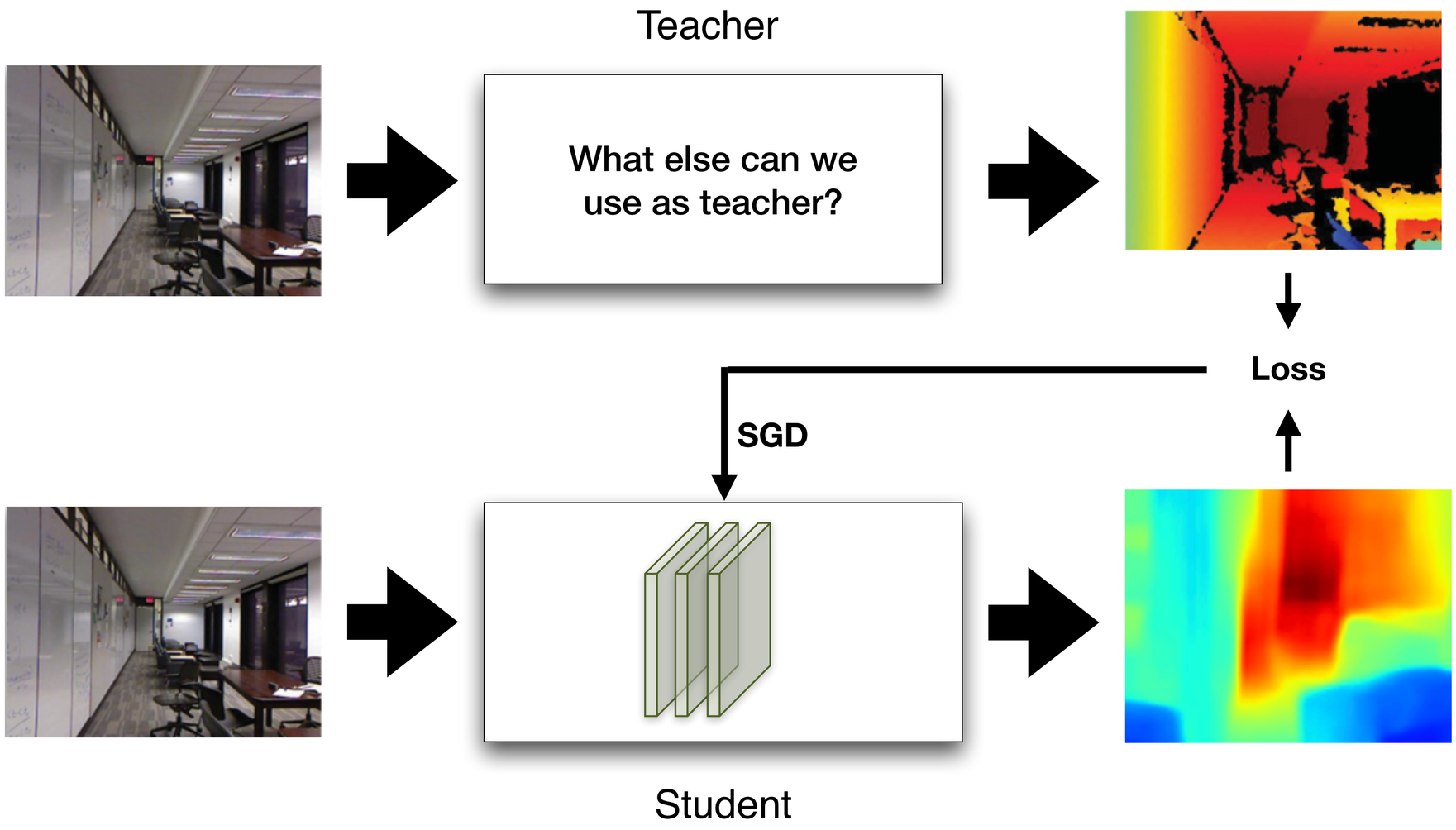
Input Prediction Ground-truth

[Eigen & Fergus, NIPS, 2014]

Kinect is a stereo active system that uses **structured light**

Ground truth is collected by using traditional methods

3D in the deep learning era



Learning Single-View Depth Prediction from Internet Photos

MegaDepth: Learning Single-View Depth Prediction from Internet Photos

Zhengqi Li Noah Snavely
Department of Computer Science & Cornell Tech, Cornell University

Abstract

Single-view depth prediction is a fundamental problem in computer vision. Recently, deep learning methods have led to significant progress, but such methods are limited by the available training data. Current datasets based on 3D sensors have key limitations, including indoor-only images (NYU), small numbers of training examples (Make3D), and sparse sampling (KITTI). We propose to use multi-view Internet photo collections, a virtually unlimited data source, to generate training data via modern structure-from-motion and multi-view stereo (MVS) methods, and present a large depth dataset called MegaDepth based on this idea. Data derived from MVS comes with its own challenges, including noise and unreconstructable objects. We address these challenges with new data cleaning methods, as well as automatically augmenting our data with ordinal depth relations generated using semantic segmentation. We validate the use of large amounts of Internet data by showing that models trained on MegaDepth exhibit strong generalization—not only to novel scenes, but also to other diverse datasets including Make3D, KITTI, and DIW, even when no images from those datasets are seen during training.[1]

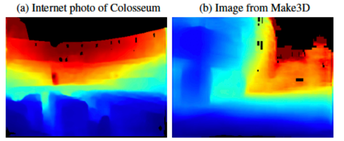
1. Introduction

Predicting 3D shape from a single image is an important capability of visual reasoning, with applications in robotics, graphics, and other vision tasks such as intrinsic images. While single-view depth estimation is a challenging, underconstrained problem, deep learning methods have recently driven significant progress. Such methods thrive when trained with large amounts of data. Unfortunately, fully general training data in the form of (*RGB image, depth map*) pairs is difficult to collect. Commodity RGB-D sensors such as Kinect have been widely used for this purpose [34], but are limited to indoor use. Laser scanners have enabled important datasets such as Make3D [29] and KITTI [25], but such devices are cumbersome to operate (in the case of industrial scanners), or produce sparse depth maps (in the case of LIDAR). Moreover, both Make3D and KITTI are collected in specific scenarios (a university campus, and atop a car, respectively). Training data can also be generated through crowdsourcing, but this approach has so far been limited to gathering sparse ordinal relationships or surface normals [12, 4, 5].

In this paper, we explore the use of a nearly unlimited source of data for this problem: images from the Internet from overlapping viewpoints, from which structure-from-

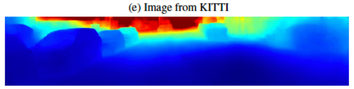
[1]Project website: http://www.cs.cornell.edu/projects/megadepth/

(a) Internet photo of Colosseum (b) Image from Make3D



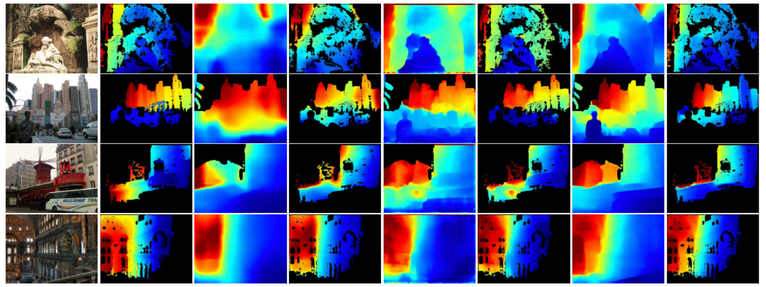
(c) Our single-view depth prediction (d) Our single-view depth prediction

(e) Image from KITTI



(f) Our single-view depth prediction

Figure 1: We use large Internet image collections, combined with 3D reconstruction and semantic labeling methods, to generate large amounts of training data for single-view depth prediction. **(a), (b), (e):** Example input RGB images. **(c), (d), (f):** Depth maps predicted by our MegaDepth-trained CNN (blue=near, red=far). For these results, the network was not trained on Make3D and KITTI data.



(a) Image (b) GT (c) VGG* (d) VGG* (M) (e) ResNet (f) ResNet (M) (g) HG (h) HG (M)

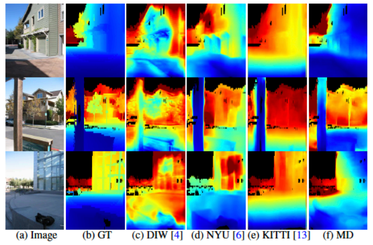
Figure 6: **Depth predictions on MD test set.** (Blue=near, red=far.) For visualization, we mask out the detected sky region. In the columns marked (M), we apply the mask from the GT depth map (indicating valid reconstructed depths) to the prediction map, to aid comparison with GT. (a) Input photo. (b) Ground truth COLMAP depth map (GT). VGG* prediction using the loss and network of [6]. (d) GT-masked version of (c). (e) Depth prediction from a ResNet [19]. (f) GT-masked version of (e). (g) Depth prediction from an hourglass (HG) network [4] . (h) GT-masked version of (g).

Training set	Method	RMS	Abs Rel	log10
Make3D	Karsch *et al.* [16]	9.20	0.355	0.127
	Liu *et al.* [24]	9.49	0.335	0.137
	Liu *et al.* [22]	8.60	0.314	0.119
	Li *et al.* [20]	7.19	0.278	0.092
	Laina *et al.* [19]	4.45	0.176	0.072
	Xu *et al.* [39]	4.38	0.184	0.065
NYU	Eigen *et al.* [6]	6.89	0.505	0.198
	Liu *et al.* [22]	7.20	0.669	0.212
	Laina *et al.* [19]	7.31	0.669	0.216
KITTI	Zhou *et al.* [43]	8.39	0.651	0.231
	Godard *et al.* [13]	9.88	0.525	0.319
DIW	Chen *et al.* [4]	7.25	0.550	0.200
MD	Ours	6.23	0.402	0.156
MD+Make3D	Ours	4.25	0.178	0.064

Table 4: **Results on Make3D for various training datasets and methods.** The first column indicates the training dataset. Errors for "Ours" are averaged over four models trained/validated on MD. Lower is better for all metrics.

depth predictions from our model and several other non-Make3D-trained models. Our network trained on MD have the best performance among all non-Make3D-trained models. Finally, the last row of Table 4 shows that our model fine-tuned on Make3D achieves better performance than the state-of-the-art.

KITTI. Next, we evaluate our model on the KITTI test set



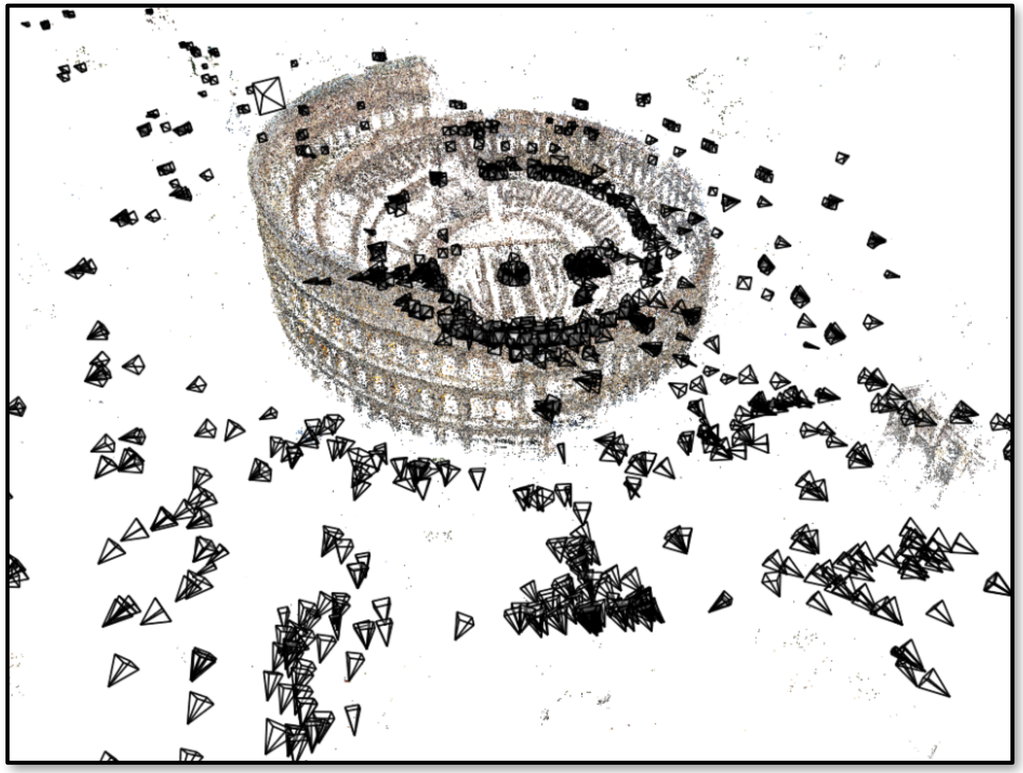
(a) Image (b) GT (c) DIW [4] (d) NYU [6] (e) KITTI [13] (f) MD

Figure 7: **Depth predictions on Make3D.** The last four columns show results from the best models trained on non-Make3D datasets (final column is our result).

based on the split of [7]. As with our Make3D experiments, we do not use images from KITTI during training. The KITTI dataset is very different from ours, consisting of driving sequences that include objects, such as sidewalks, cars, and people, that are difficult to reconstruct with SfM/MVS. Nevertheless, as shown in Table 5, our MD-trained network still outperform approaches trained on non-KITTI datasets. Finally, the last row of Table 5 shows that we can achieve state-of-the-art performance by fine-tuning our network on KITTI training data. Figure 8 shows visual comparisons between our results and models trained on other non-KITTI

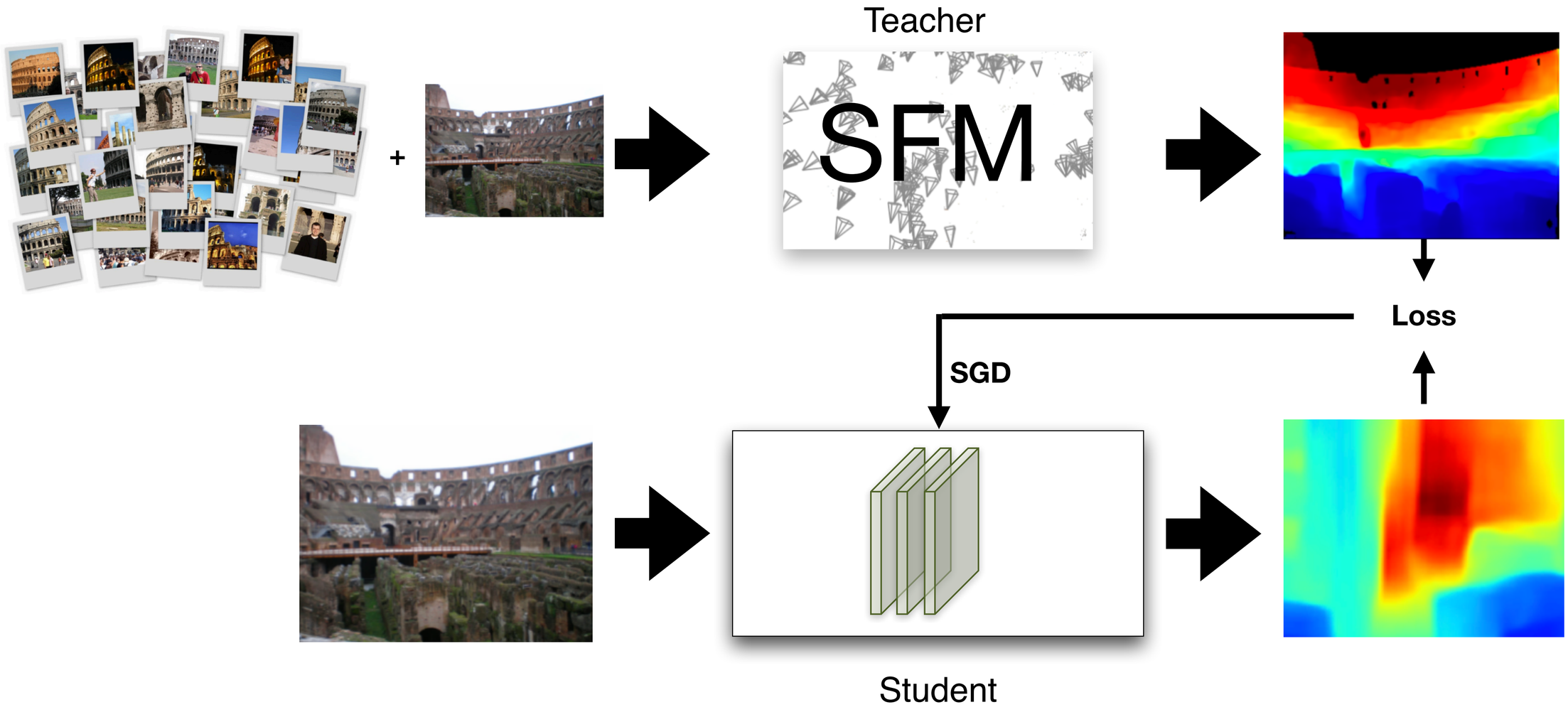
CVPR 2018

Structure from motion (stereo but with many cameras)



The internet can be an unlimited source of 3D data

Structure from motion



MegaDepth dataset

200 locations, ~130k images

http://www.cs.cornell.edu/projects/megadepth

Stacked hourglass architecture

Stacked Hourglass Networks for Human Pose Estimation

Alejandro Newell, Kaiyu Yang, and Jia Deng

University of Michigan, Ann Arbor
{alnewell,yangky,jiadeng}@umich.edu

Abstract. This work introduces a novel convolutional network architecture for the task of human pose estimation. Features are processed across all scales and consolidated to best capture the various spatial relationships associated with the body. We show how repeated bottom-up, top-down processing used in conjunction with intermediate supervision is critical to improving the performance of the network. We refer to the architecture as a "stacked hourglass" network based on the successive steps of pooling and upsampling that are done to produce a final set of predictions. State-of-the-art results are achieved on the FLIC and MPII benchmarks outcompeting all recent methods.

Keywords: Human Pose Estimation

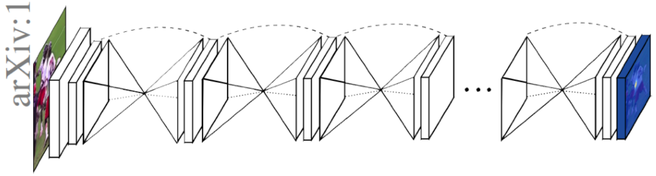
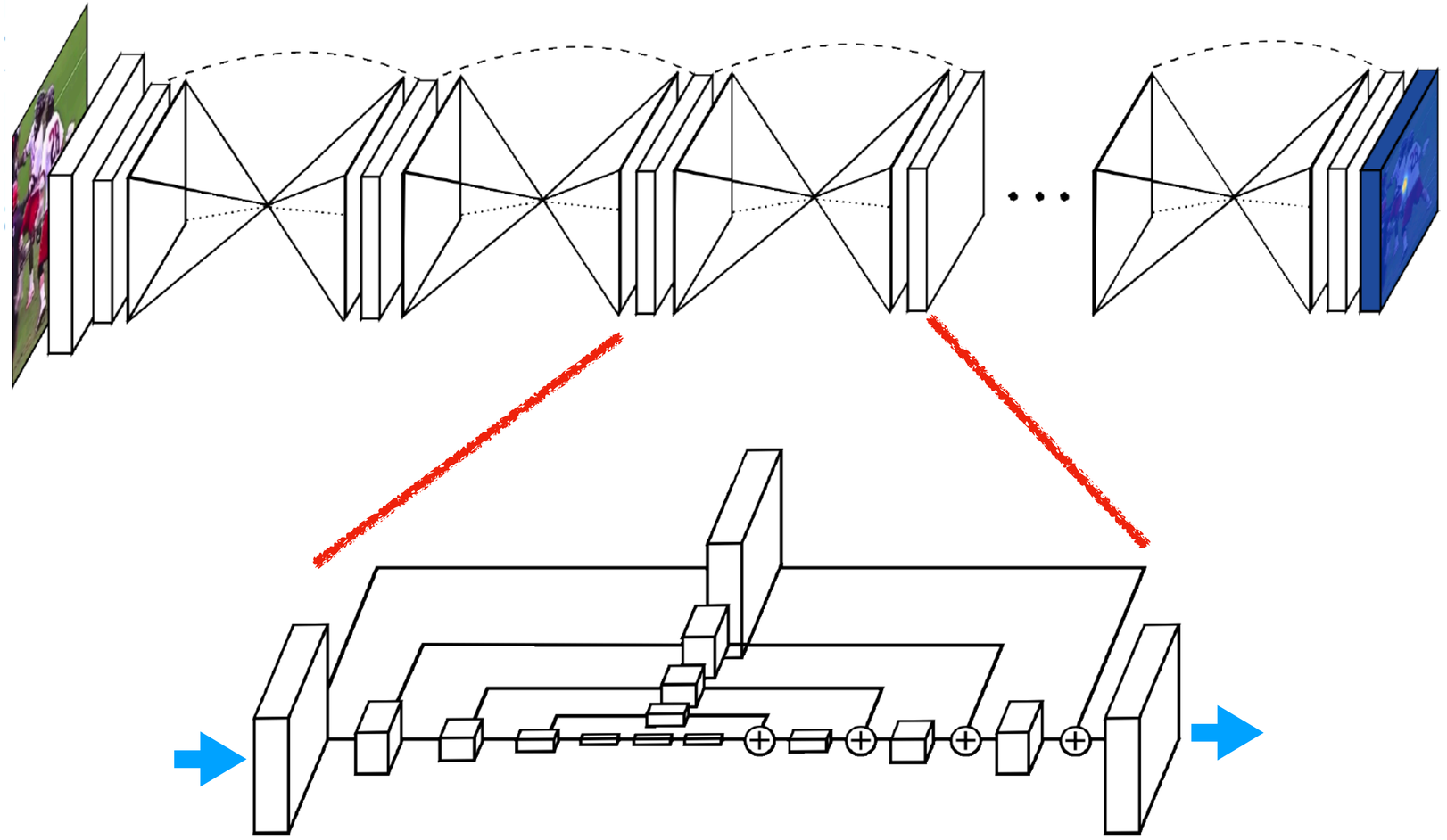


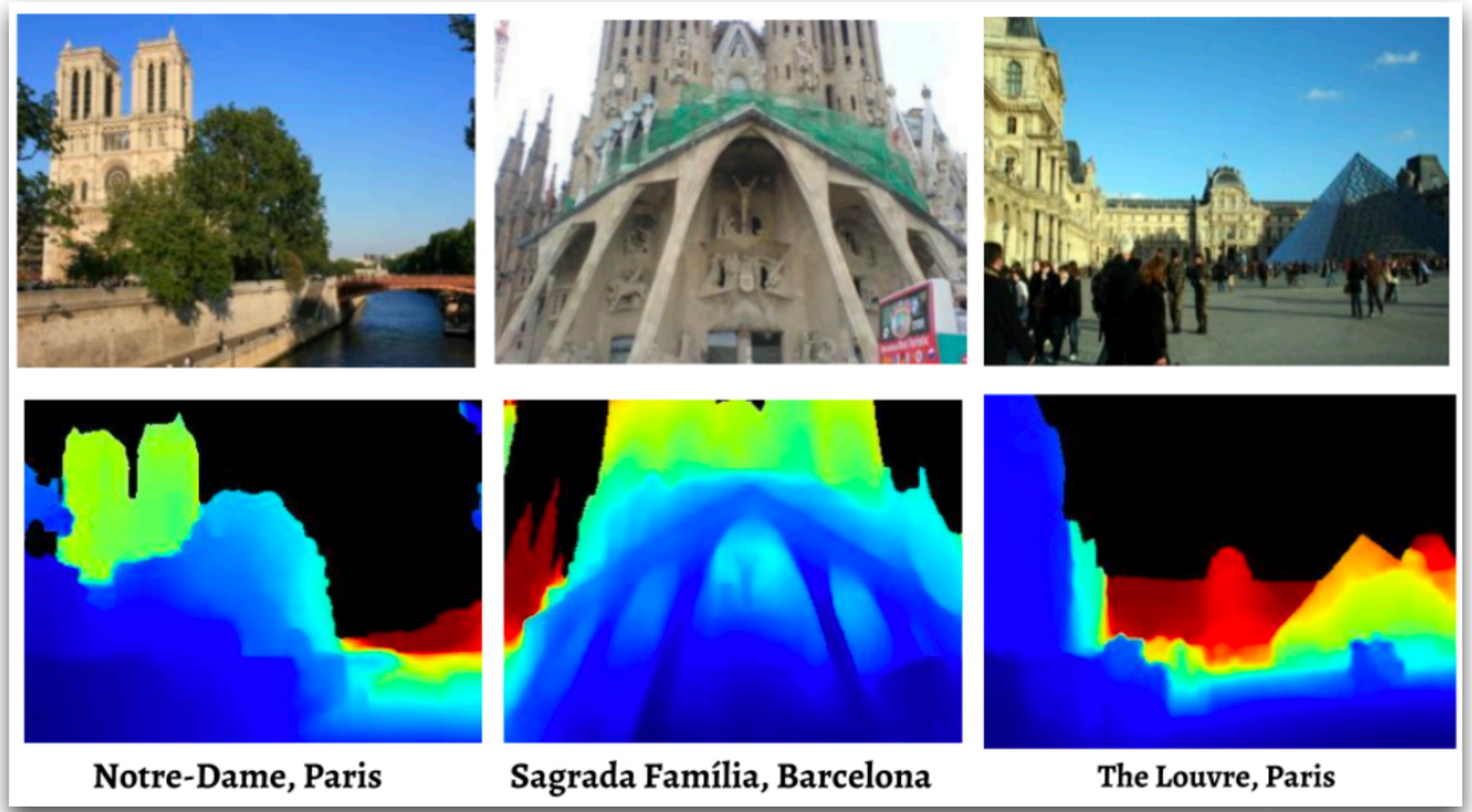
Fig. 1. Our network for pose estimation consists of multiple stacked hourglass modules which allow for repeated bottom-up, top-down inference.

1 Introduction

A key step toward understanding people in images and video is accurate pose estimation. Given a single RGB image, we wish to determine the precise pixel location of important keypoints of the body. Achieving an understanding of a person's posture and limb articulation is useful for higher level tasks like action recognition, and also serves as a fundamental tool in fields such as human-computer interaction and animation.

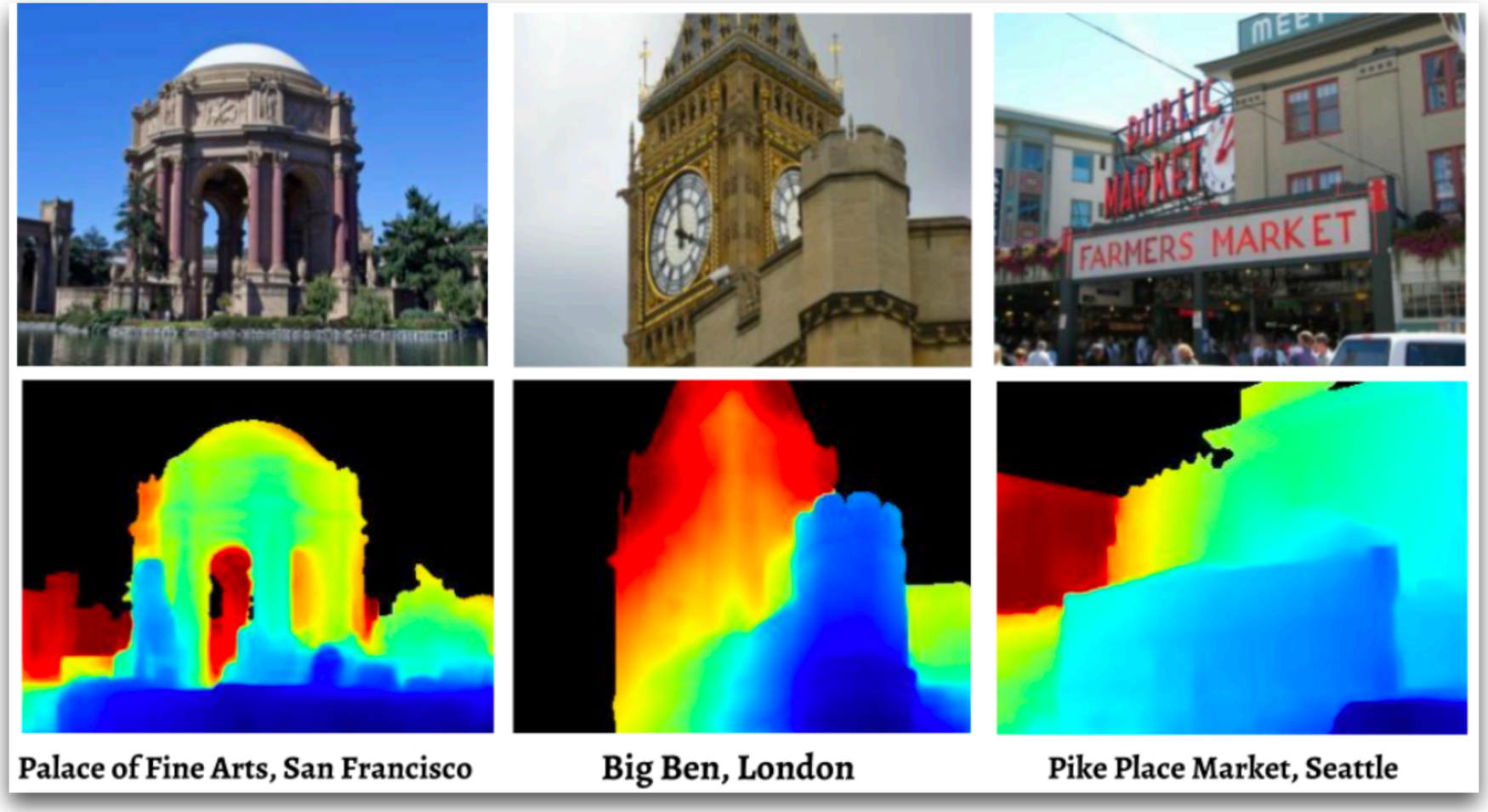


MegaDepth results



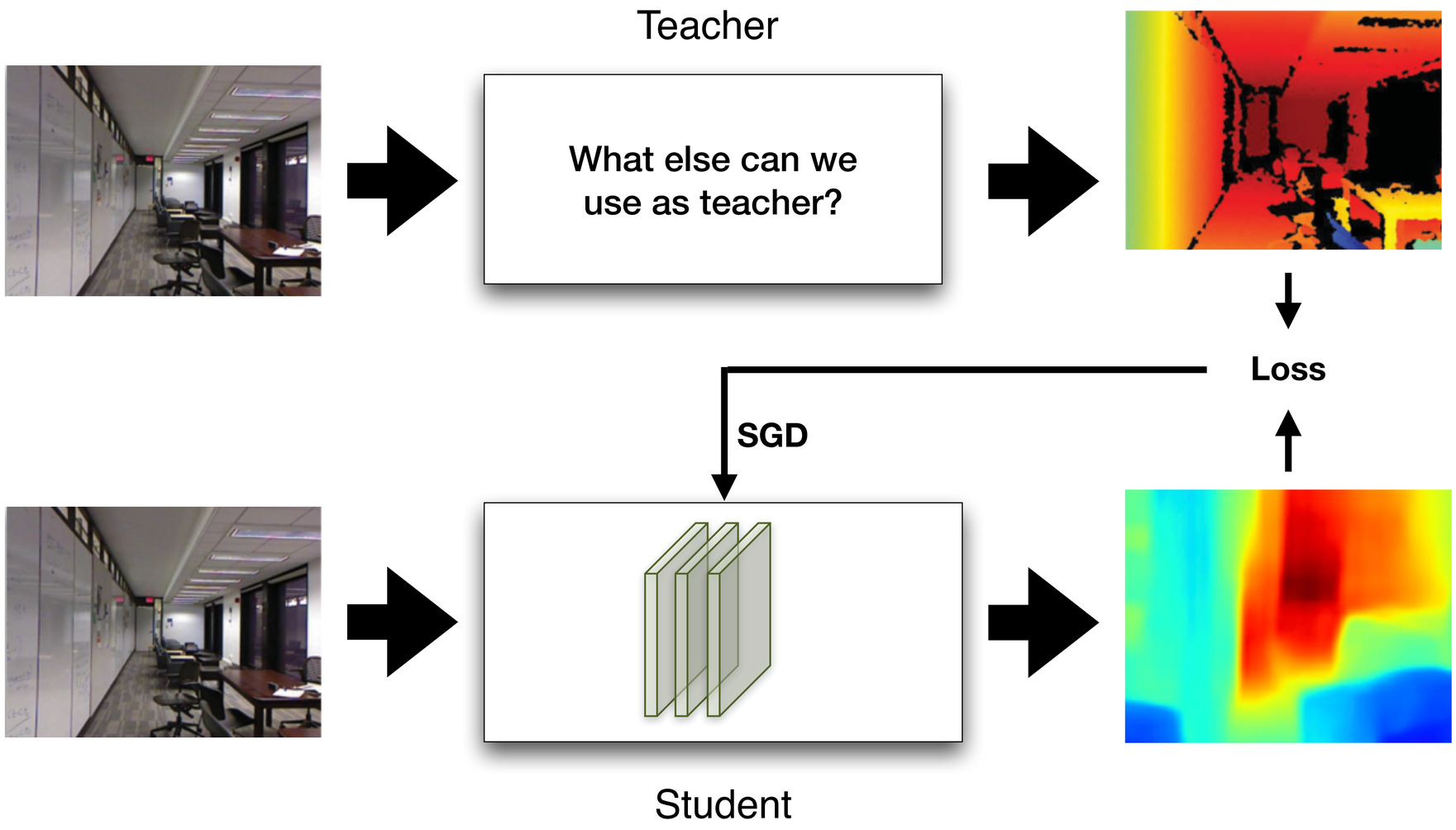
Source: http://www.cs.cornell.edu/projects/megadepth/

MegaDepth results



Source: http://www.cs.cornell.edu/projects/megadepth/

3D in the deep learning era



How else can we collect depth?

“Unsupervised” camera motion and monocular depth

Unsupervised Learning of Depth and Ego-Motion from Video

Tinghui Zhou* UC Berkeley · Matthew Brown Google · Noah Snavely Google · David G. Lowe Google

Abstract

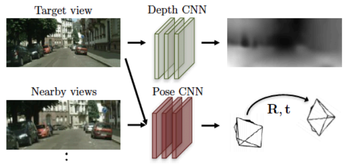
We present an unsupervised learning framework for the task of monocular depth and camera motion estimation from unstructured video sequences. In common with recent work [10, 14, 16], we use an end-to-end learning approach with view synthesis as the supervisory signal. In contrast to the previous work, our method is completely unsupervised, requiring only monocular video sequences for training. Our method uses single-view depth and multi-view pose networks, with a loss based on warping nearby views to the target using the computed depth and pose. The networks are thus coupled by the loss during training, but can be applied independently at test time. Empirical evaluation on the KITTI dataset demonstrates the effectiveness of our approach: 1) monocular depth performs comparably with supervised methods that use either ground-truth pose or depth for training, and 2) pose estimation performs favorably compared to established SLAM systems under comparable input settings.

1. Introduction

Humans are remarkably capable of inferring ego-motion and the 3D structure of a scene even over short timescales. For instance, in navigating along a street, we can easily locate obstacles and react quickly to avoid them. Years of research in geometric computer vision has failed to recreate similar modeling capabilities for real-world scenes (e.g., where non-rigidity, occlusion and lack of texture are present). So why do humans excel at this task? One hypothesis is that we develop a rich, structural understanding of the world through our past visual experience that has largely consisted of moving around and observing vast numbers of scenes and developing *consistent* modeling of our observations. From millions of such observations, we have learned about the regularities of the world—roads are flat, buildings are straight, cars are supported by roads *etc.*, and we can apply this knowledge when perceiving a new scene, even from a single monocular image.

*The majority of the work was done while interning at Google.

(a) Training: unlabeled video clips.

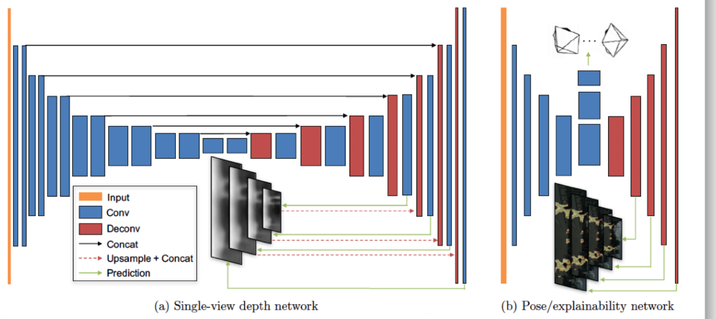


(b) Testing: single-view depth and multi-view pose estimation.

Figure 1. The training data to our system consists solely of unlabeled image sequences capturing scene appearance from different viewpoints, where the poses of the images are not provided. Our training procedure produces two models that operate independently, one for single-view depth prediction, and one for multi-view camera pose estimation.

In this work, we mimic this approach by training a model that observes sequences of images and aims to explain its observations by predicting likely camera motion and the scene structure (as shown in Fig. 1). We take an end-to-end approach in allowing the model to map directly from input pixels to an estimate of ego-motion (parameterized as 6-DoF transformation matrices) and the underlying scene structure (parameterized as per-pixel depth maps under a reference view). We are particularly inspired by prior work that has suggested view synthesis as a metric [44] and recent work that tackles the calibrated, multi-view 3D case in an end-to-end framework [10]. Our method is unsupervised, and can be trained simply using sequences of images with no manual labeling or even camera motion information.

Our approach builds upon the insight that a geometric view synthesis system only performs *consistently* well when its intermediate predictions of the scene geometry and the camera poses correspond to the physical ground-



(a) Single-view depth network (b) Pose/explainability network

Figure 4. Network architecture for our depth/pose/explainability prediction modules. The width and height of each rectangular block indicates the output channels and the spatial dimension of the feature map at the corresponding layer respectively, and each reduction/increase in size indicates a change by the factor of 2. (a) For single-view depth, we adopt the DispNet [35] architecture with multi-scale side predictions. The kernel size is 3 for all the layers except for the first 4 conv layers with 7, 7, 5, 5, respectively. The number of output channels for the first conv layer is 32. (b) The pose and explainabilty networks share the first few conv layers, and then branch out to predict 6-DoF relative pose and multi-scale explainability masks, respectively. The number of output channels for the first conv layer is 16, and the kernel size is 3 for all the layers except for the first two conv and the last two deconv/prediction layers where we use 7, 5, 5, 7, respectively. See Section 3.5 for more details.

network's belief in where direct view synthesis will be successfully modeled for each target pixel. Based on the predicted $\hat{E}_s$, the view synthesis objective is weighted correspondingly by

$$\mathcal{L}_{vs} = \sum_{<I_1,...,I_N>\in\mathcal{S}} \sum_{p} \hat{E}_s(p) |I_t(p) - \hat{I}_s(p)| \,. \quad (3)$$

Since we do not have direct supervision for $\hat{E}_s$, training with the above loss would result in a trivial solution of the network always predicting $\hat{E}_s$ to be zero, which perfectly minimizes the loss. To resolve this, we add a regularization term $\mathcal{L}_{reg}(\hat{E}_s)$ that encourages nonzero predictions by minimizing the cross-entropy loss with constant label 1 at each pixel location. In other words, the network is encouraged to minimize the view synthesis objective, but allowed a certain amount of slack for discounting the factors not considered by the model.

3.4. Overcoming the gradient locality

One remaining issue with the above learning pipeline is that the gradients are mainly derived from the pixel intensity difference between $I(p_t)$ and the four neighbors of $I(p_s)$, which would inhibit training if the correct p_s (projected using the ground-truth depth and pose) is located in a low-texture region or far from the current estimation. This is a well known issue in motion estimation [3]. Empirically, we found two strategies to be effective for overcoming this issue: 1) using a convolutional encoder-decoder architecture with a small bottleneck for the depth network that implicitly constrains the output to be globally smooth and facilitates gradients to propagate from meaningful regions to nearby regions; 2) explicit multi-scale and smoothness loss (e.g., as in [14, 16]) that allows gradients to be derived from larger spatial regions directly. We adopt the second strategy in this work as it is less sensitive to architectural choices. For smoothness, we minimize the L_1 norm of the second-order gradients for the predicted depth maps (similar to [48]).

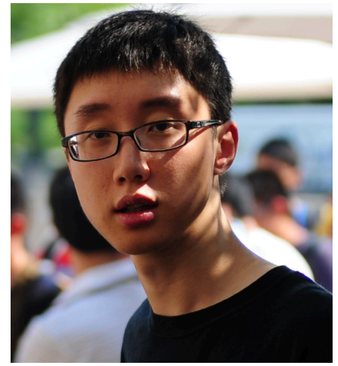
Our final objective becomes

$$\mathcal{L}_{final} = \sum_{l} \mathcal{L}^l_{vs} + \lambda_s \mathcal{L}^l_{smooth} + \lambda_e \sum_{s} \mathcal{L}_{reg}(\hat{E}^l_s) \,, \quad (4)$$

where l indexes over different image scales, s indexes over source images, and λ_s and λ_e are the weighting for the depth smoothness loss and the explainability regularization, respectively.

3.5. Network architecture

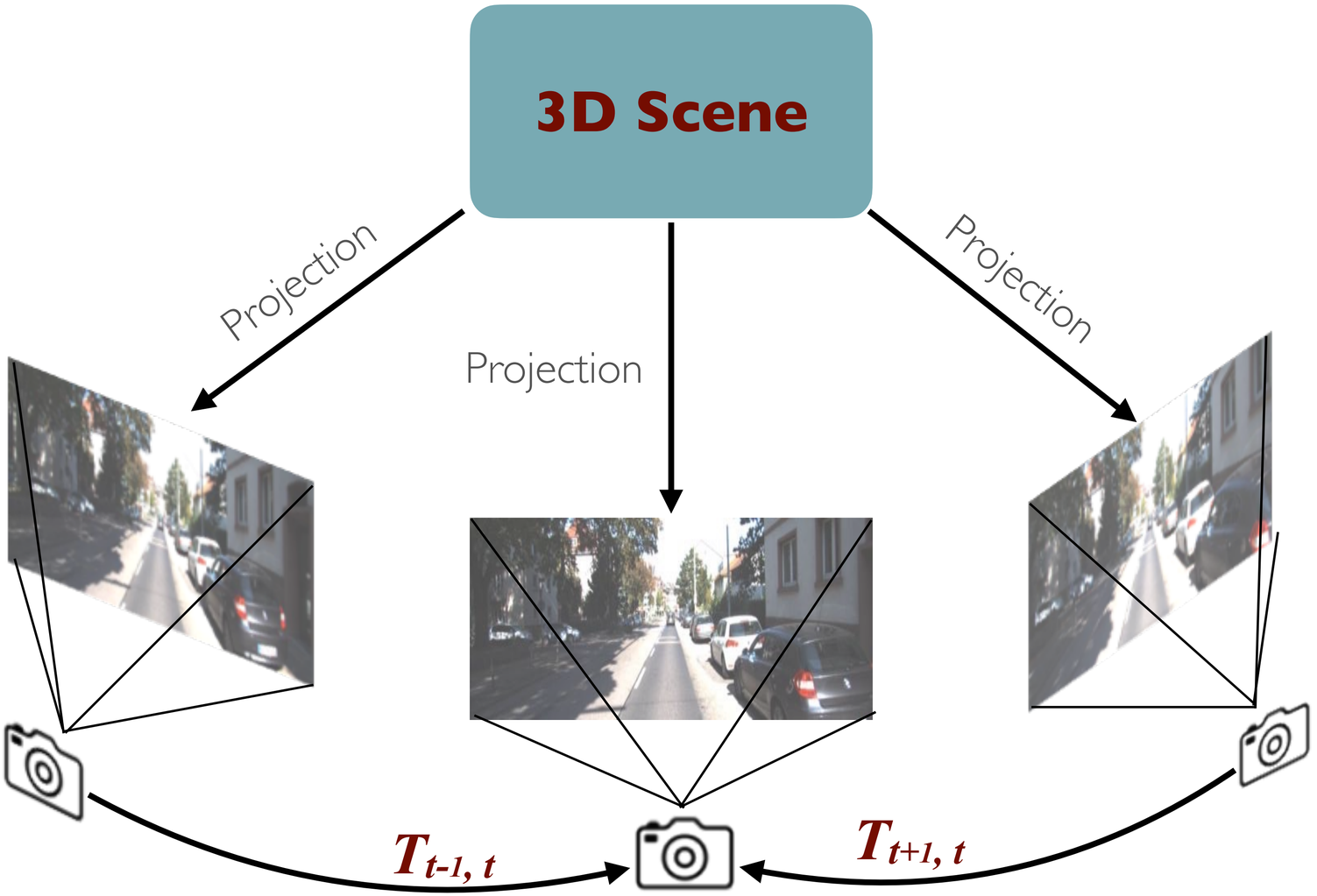
Single-view depth For single-view depth prediction, we adopt the DispNet architecture proposed in [35] that is mainly based on an encoder-decoder design with skip connections and multi-scale side predictions (see Figure 4). All conv layers are followed by ReLU activation except for the prediction layers, where we use $1/(\alpha * sigmoid(x) + \beta)$ with $\alpha = 10$ and $\beta = 0.1$ to constrain the predicted depth to be always positive within a reasonable range. We also experimented with using multiple views as input to the depth network, but did not find this to improve the results. This is in line with the observations in [47], where optical flow constraints need to be enforced to utilize multiple views effectively.



Tinghui Zhou

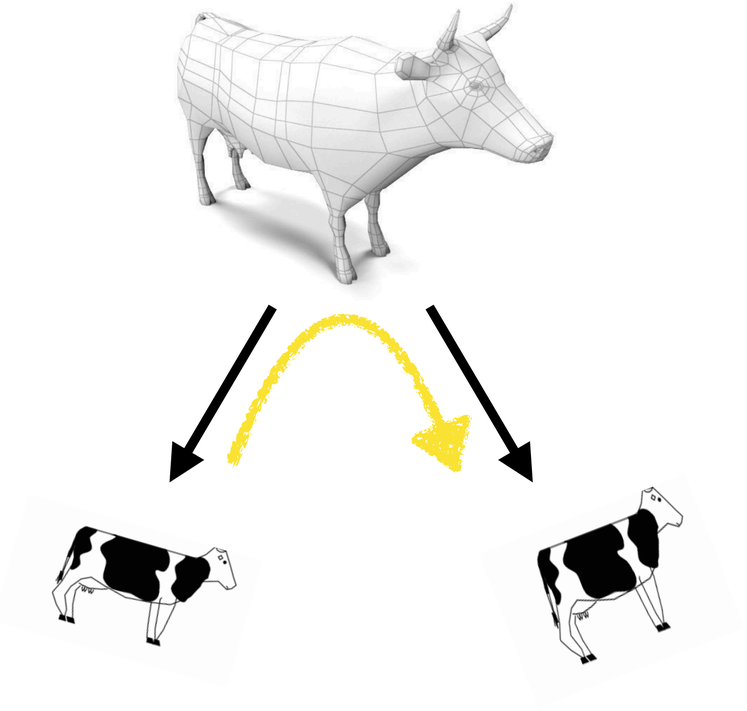
CVPR 2017

Main idea: supervision by view synthesis



Source: Tinghui Zhou

The allegory of the cave



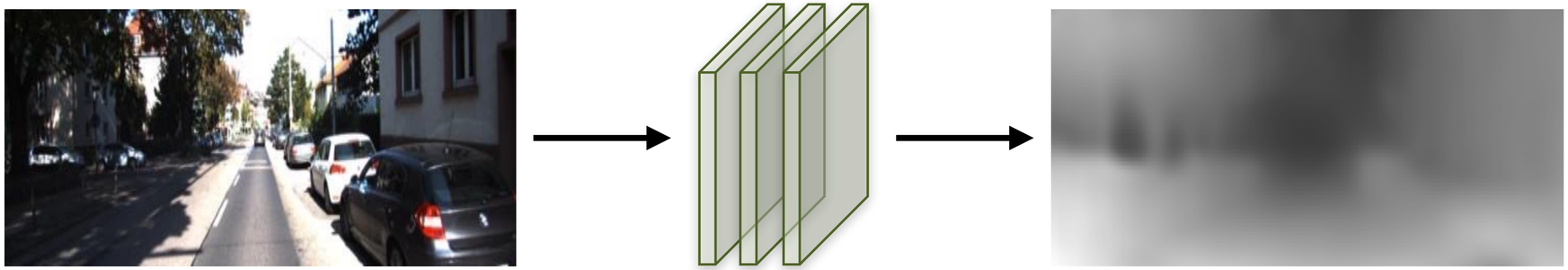
System components

Depth estimation

Pose estimation

View synthesis

I_t

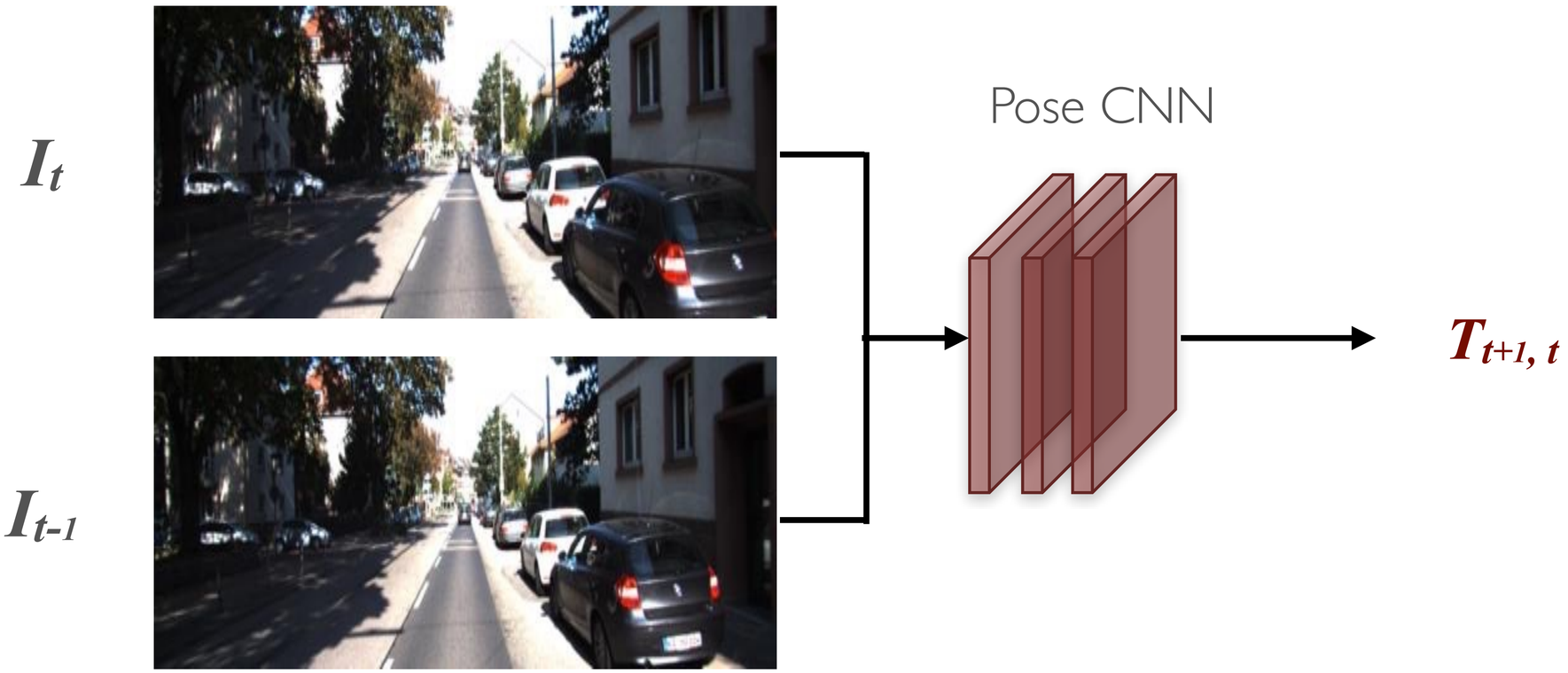


System components

Depth estimation

Pose estimation

View synthesis

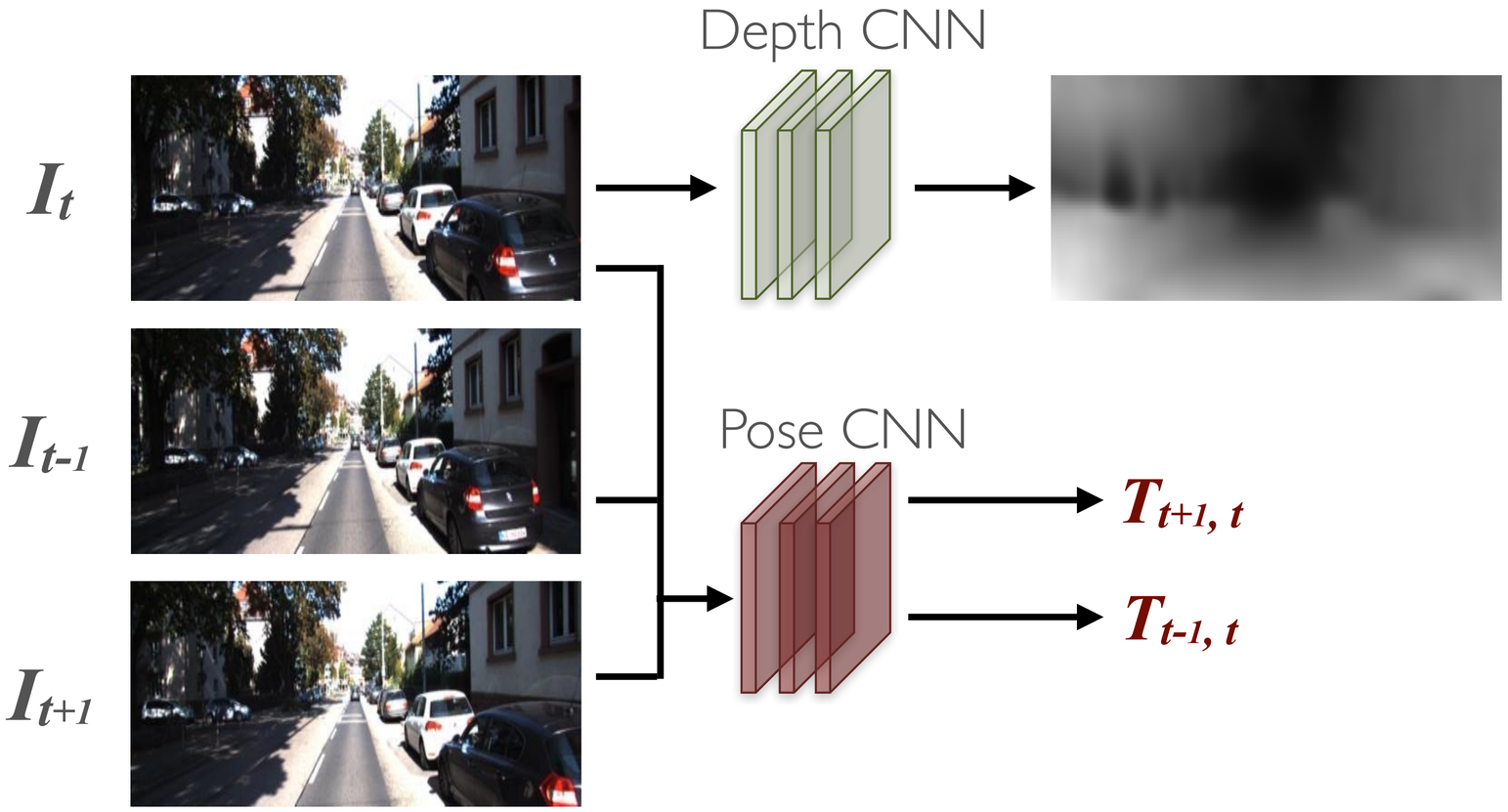


System components

Depth estimation

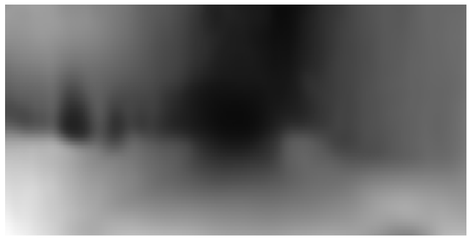
Pose estimation

View synthesis



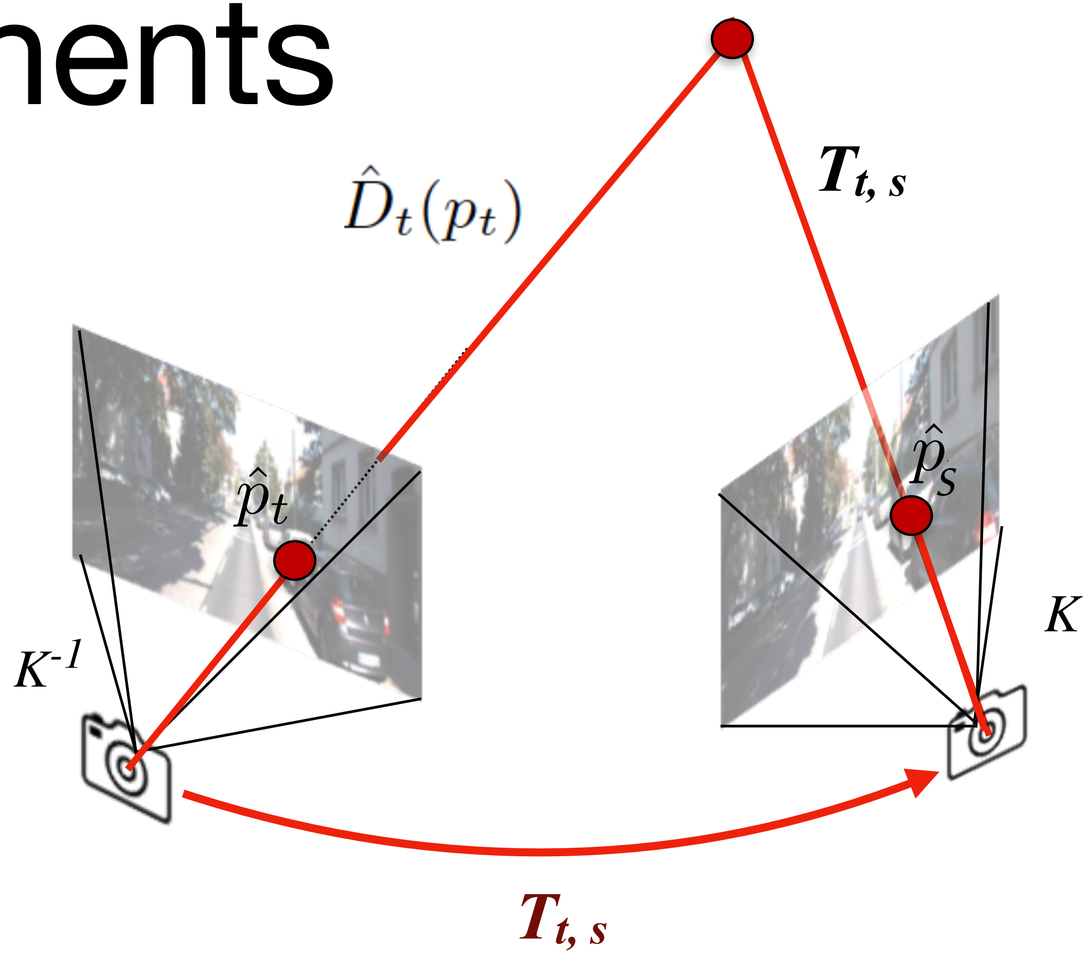
System components

Depth estimation



Pose estimation $T_{t,\,s}$

View synthesis



$$p_s \sim K \hat{T}_{t \to s} \hat{D}_t(p_t) K^{-1} p_t$$

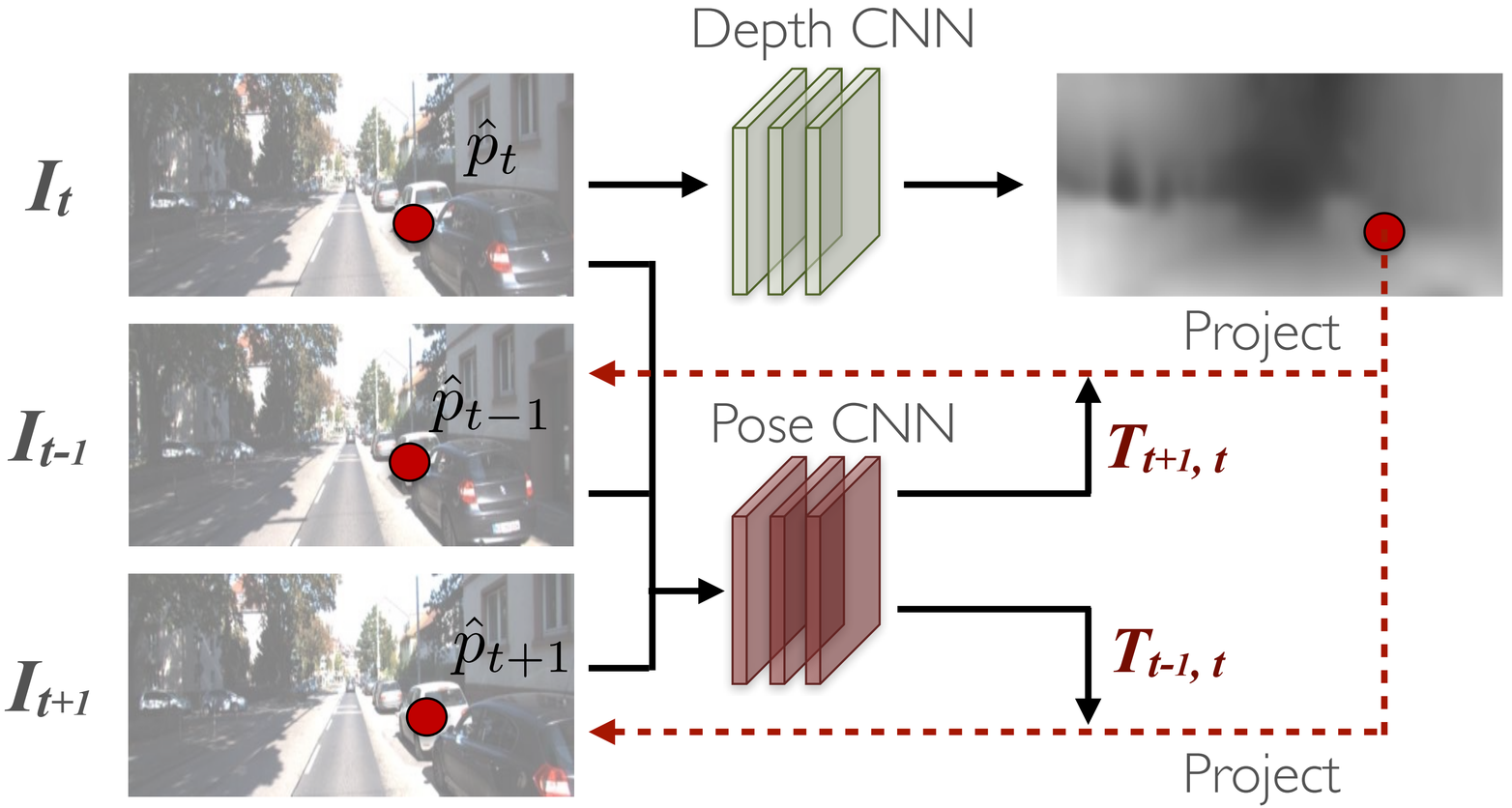
In this work, K is assumed to be known!

System components

Depth estimation

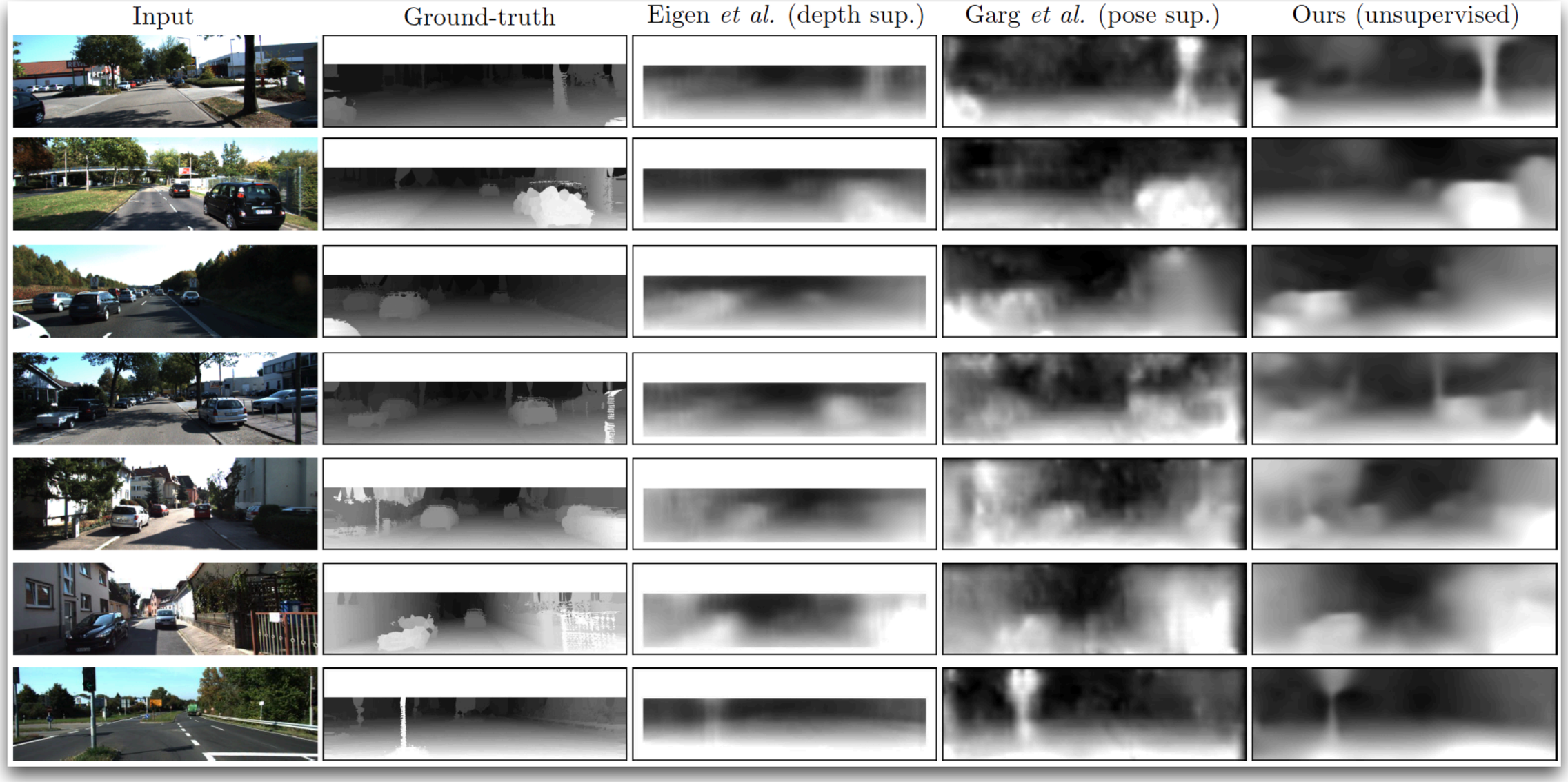
Pose estimation

View synthesis



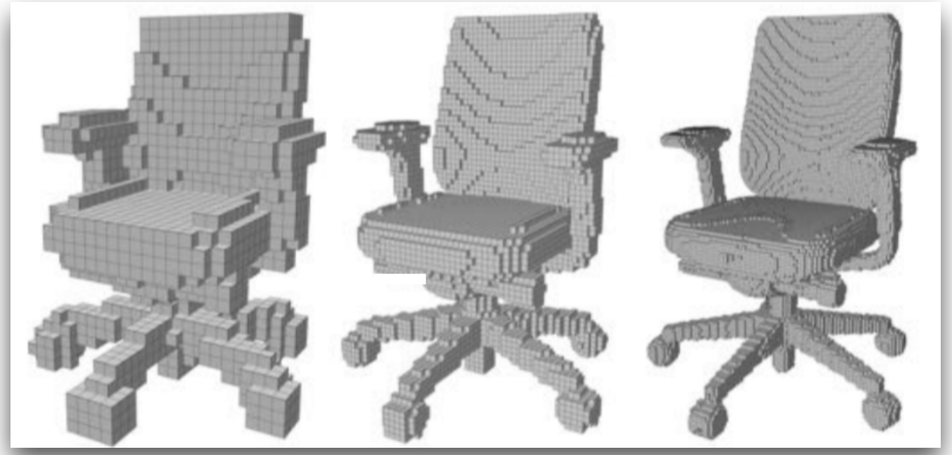
Training Loss: $\mathcal{L}_{vs} = \sum_{s \in \{\text{nearby frames}\}} \sum_{p} |I_t(p_t) - I_s(\hat{p}_s)|$

Results

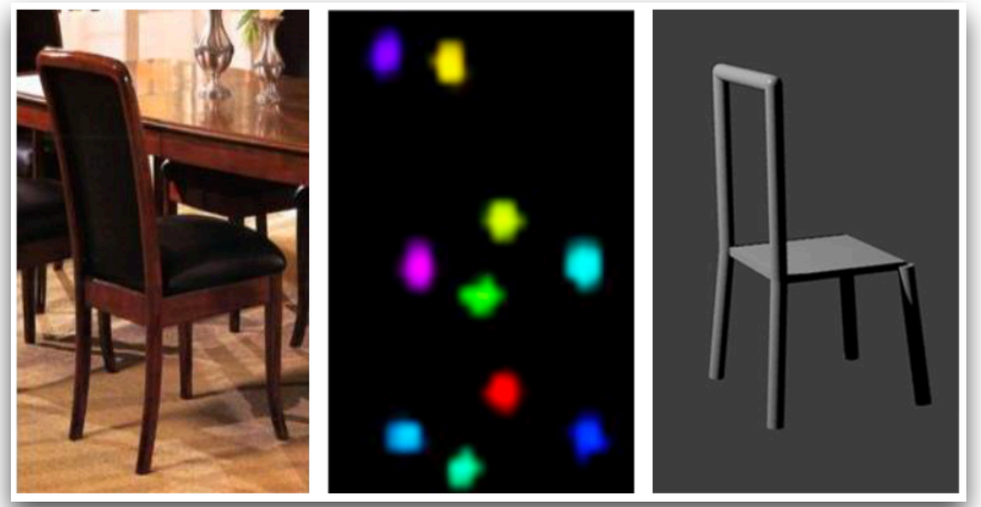

Input
Ground-truth
Eigen *et al.* (depth sup.)
Garg *et al.* (pose sup.)
Ours (unsupervised)

3D Object Models

Voxels



3D Keypoint



Multiviews

Meshes

15. Scene Understanding

- **Semantics**
  - Object detection
  - Semantic segmentation
  - Instance segmentation
- **Geometry**
  - 3D in the deep learning era
  - Single view depth estimation
  - Unsupervised learning of monocular depth cues