Towards Robust Scene Understanding

Dr. Lokender Tiwari

Research Scientist

Outline

- Introduction
- Need for robustness in scene understanding
- Towards robust scene understanding
    - Geometric Scene Understanding
        - Visual SLAM Basics
        - Self-Improving geometric-CNN Framework for 3D Perception
    - Semantic Scene Understanding
        - Robust Image Classification
- 3D Virtual Try-on for Metaverse / ECommerce

Introduction

- A scene can contain multiple physical objects like people, vehicles interacting with each other or with the environment

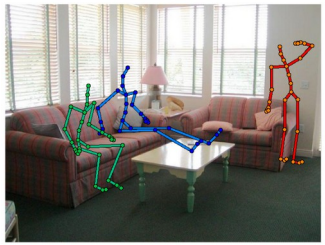
(a) (b)

Fig . Example of an (a) outdoor scene, (b) indoor scene

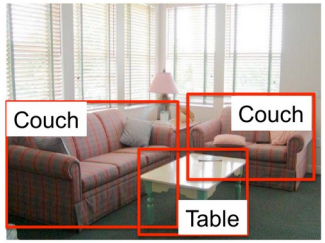
Introduction

Autonomous agent

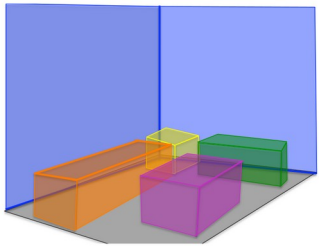
Environment



Interaction



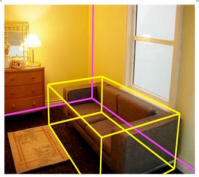
Scene semantic



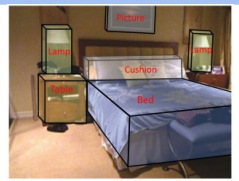
Scene geometry

3D Map of the environment

Scene layout



Objects in the scene



Spatial Location of Objects

.....

+ many more

- Semantic segmentation
- Image captioning
- ...

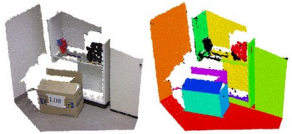
Scene Understanding

[1] Gupta, et al. "From 3d scene geometry to human workspace." *CVPR 2011*. IEEE, 2011.
[2] Naseer et al. “Indoor Scene Understanding in 2.5/3D: A Survey”. IEEE Access 2018
[3] Yao et al. “Describing the Scene as a Whole: Joint Object Detection, Scene Classification and Semantic Segmentation”, CVPR, 2012.

Need for Robustness

- Applications like autonomous driving, service robotics
  - relies on both geometric and semantic information
  - multiple task specific estimators or predictors e.g., depth estimator, object detector, classifiers
  - performance depends on the robustness of these estimators
- Estimators can make errors due to:
  - noise and outliers in the input, challenging scenarios, violation of modeling assumptions
- Context of noise and outliers differs for different task specific estimators

Plane Fitting

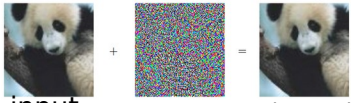


Noise : Depth sensor noise
Outliers: Non planar points

Monocular Depth Prediction [1]

Self-supervised depth models: biased depth estimates due to brightness constancy
Outliers: moving objects

Image Classification



Noise: Adversarial noise

[1] Godard et.al, Digging into Self-Supervised Monocular Depth Prediction, ICCV, 2019

What is a Visual-SLAM?

Visual
Simultaneous
Locatization
And
Mapping

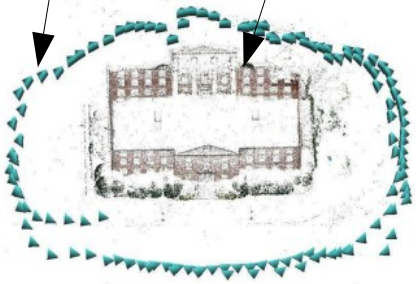
Vision Based Sensors (Monocular, Stereo, RGB-D)

simultaneous estimation of the state (**position and orientation**) of an autonomous agent with on-board sensors, and the construction of a model (**map**) of the environment that the sensors are perceiving.

RGB Images

Camera (Position and Orientation) **Known**

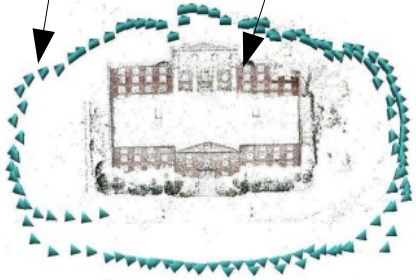
Map



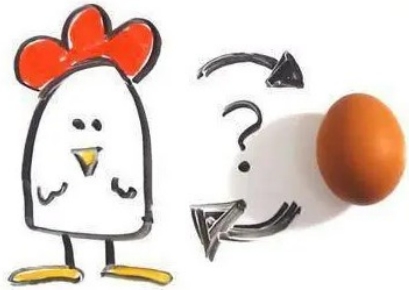
Construct the Map

Camera (Position and Orientation)

Map **Known**



Estimate the camera pose



Applications

Autonomous Driving

Augmented Reality

Virtual Reality

Configurations

Monocular Vision

Binocular Vision

Trinocular Vision

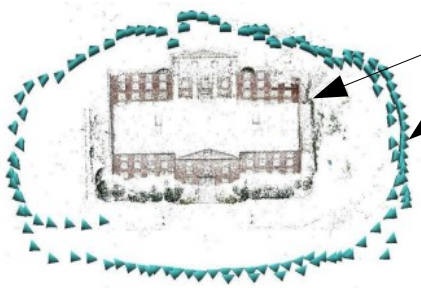
Monocular + LiDAR

Stereo + LiDAR

RGB-D Camera

RGB-D + Stereo

Visual SLAM : Basics



$\mathbf{x}_{wj} \in \mathbb{R}^3$ Coordinates of point j

$\mathbf{T}_{iw} \in \mathrm{SE}(3)$ Pose of camera i

Unknowns

$\mathbf{u}_{ij} = \begin{bmatrix} u_{ij} \\ v_{ij} \end{bmatrix}$ Observation of point j from camera i

Known

Find $\mathbf{T}_{iw}$ and $\mathbf{x}_{wj}$ by minimizing the reprojection errors

$$\mathbf{e}_{ij} = \mathbf{u}_{ij} - \pi_i(\mathbf{T}_{iw}, \mathbf{x}_{wj})$$

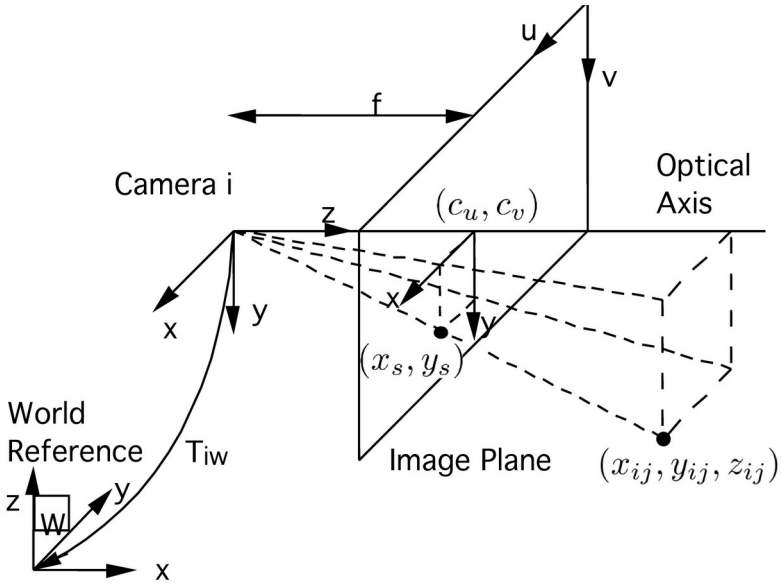
Re-projection error

Projection Function

$$\mathbf{T}_{iw} \in \mathrm{SE}(3) = [\mathbf{R}_{iw}, \mathbf{t}_{iw}], \quad \mathbf{R}_{iw} \in \mathrm{SO}(3), \mathbf{t}_{iw} \in \mathbb{R}^3$$

$$\mathbf{x}_{ij} = \mathbf{R}_{iw}\mathbf{x}_{wj} + \mathbf{t}_{iw}$$

Visual SLAM : Basics



focal length (mm)

$$x_s = f_i \frac{x_{ij}}{z_{ij}}$$

$$u = (s_u f_i)\frac{x_{ij}}{z_{ij}} + c_{i,u}$$

$$= f_{i,u}\frac{x_{ij}}{z_{ij}} + c_{i,u}$$

horizontal focal length (pixels)

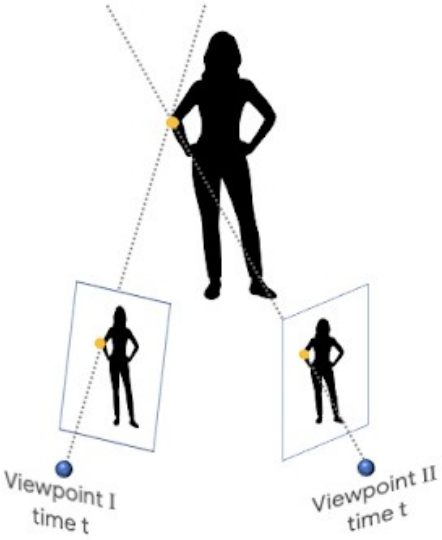
principal point

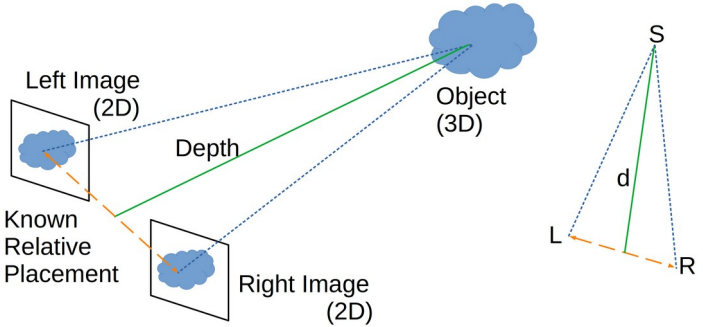
- In summary:

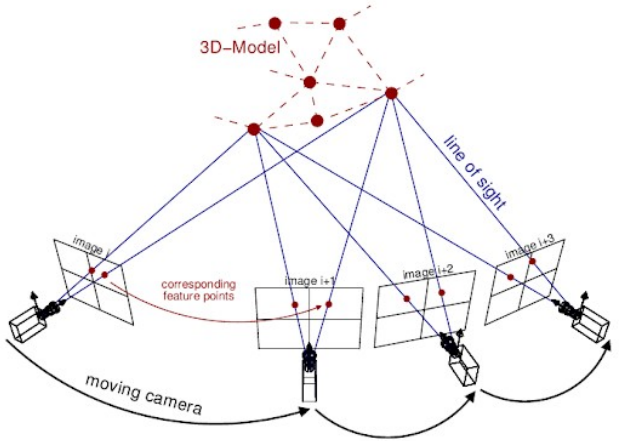
$$\pi_i(\mathbf{T}_{iw}, \mathbf{x}_{wj}) = \begin{bmatrix} f_{i,u}\frac{x_{ij}}{z_{ij}} + c_{i,u} \\ f_{i,v}\frac{y_{ij}}{z_{ij}} + c_{i,v} \end{bmatrix}$$

Visual SLAM : Basics

- Triangulation
- Bundle Adjustment







Visual SLAM : Basics

Find $\mathbf{T}_{iw}$ and $\mathbf{x}_{wj}$ by minimizing the reprojection errors

$$\mathbf{e}_{ij} = \mathbf{u}_{ij} - \pi_i(\mathbf{T}_{iw}, \mathbf{x}_{wj})$$

Reprojection error

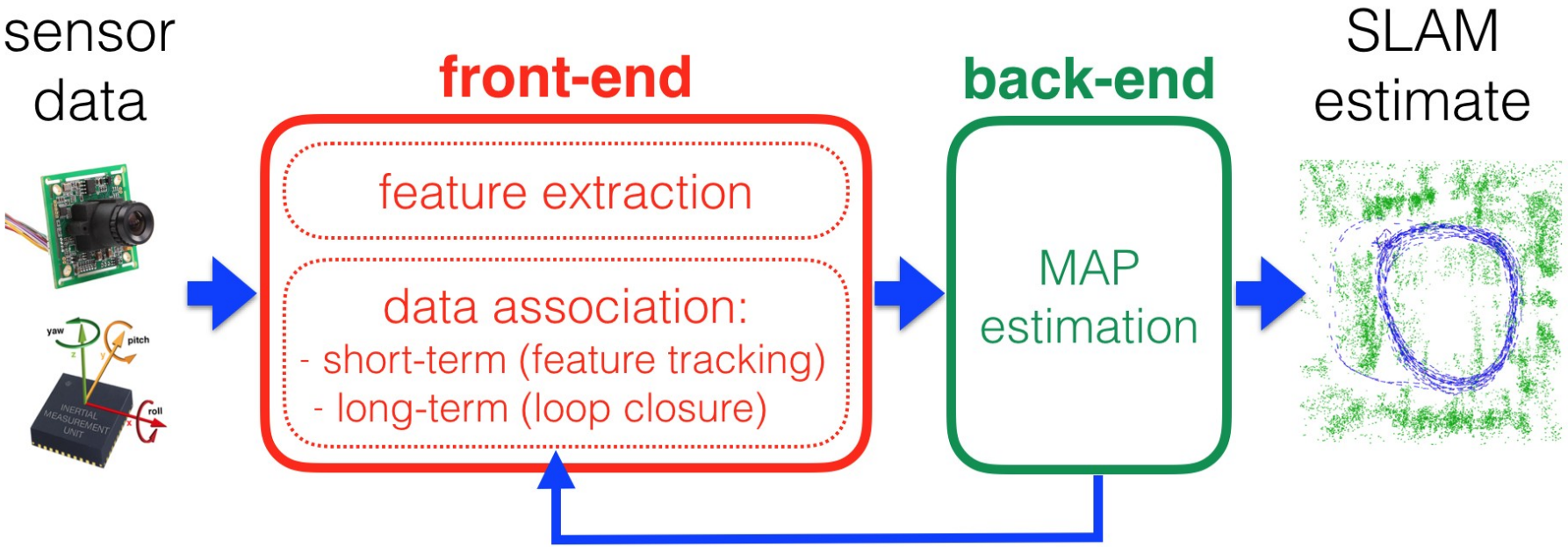
Projection Function

Cost Function

$$\{\mathbf{T}_{1w}..\mathbf{T}_{nw}, \mathbf{x}_{w1}..\mathbf{x}_{wm}\}^\star = \arg\min_{\mathbf{T},\mathbf{x}} \sum_{i,j} \rho_h(\mathbf{e}_{ij}^T \mathbf{\Sigma}_{ij}^{-1} \mathbf{e}_{ij})$$

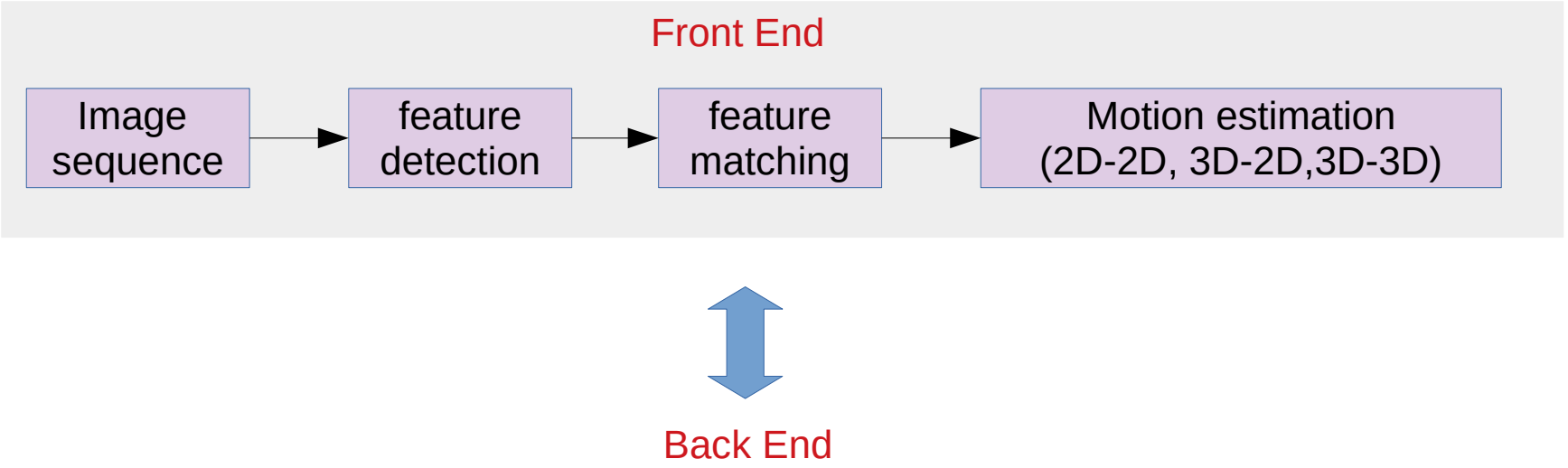
Bundle Adjustment

A Typical Visual SLAM system



Cadena, Cesar, et al. "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age." IEEE Transactions on robotics 32.6 (2016):

Visual SLAM system : Front-End

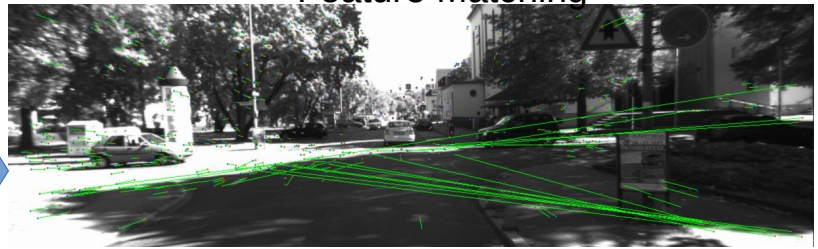


Visual SLAM system : Front-End

Image Sequence

Feature Detection

Feature Matching

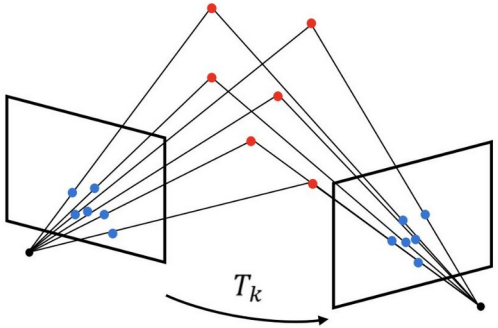


Outlier Rejection and Pose Estimation

Visual SLAM system : Front-End

Motion Estimation : **2D-2D** Essential Matrix

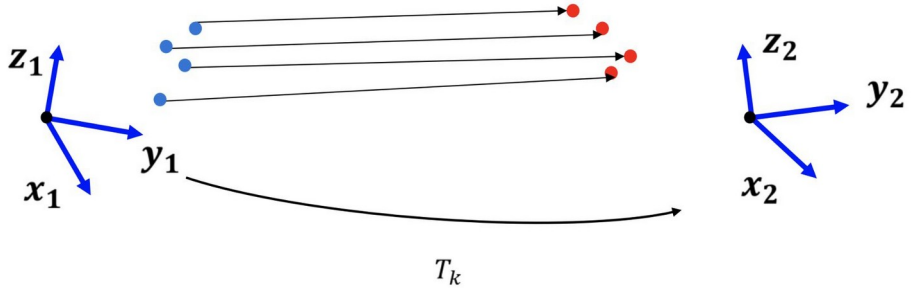
- Minimize reprojection error
- Impossible if the camera purely rotates



Motion Estimation : **3D-3D** Iterative Closest Point (ICP)

- Given two sets of 3D points, iteratively estimate the transformation
  T that minimizes the distance between points

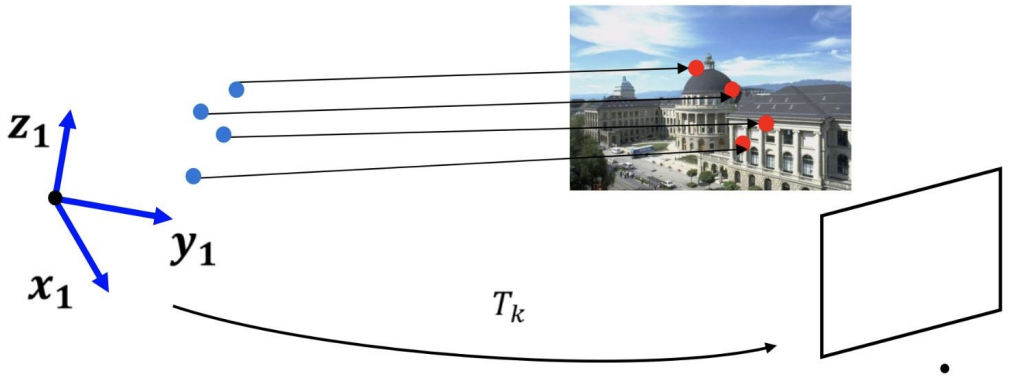
$$T_k = \begin{bmatrix} R_{k,k-1} & t_{k,k-1} \\ 0 & 1 \end{bmatrix} = \arg\min_{T_k} \sum_i ||\tilde{X}_k^i - T_k \tilde{X}_{k-1}^i||$$



Visual SLAM system : Front-End

Motion Estimation : **3D-2D** Perspective from n Points (PnP)

- Obtained by determining the transformation that minimizes the reprojection error.



Visual SLAM system : Back-End

Camera Pose Optimization
- Optimize **all** the camera rotations and translations)

Bundle Adjustment[1]
- Jointly optimize all the camera poses and the position of 3D points

Loop Closure
- Jointly optimize all the camera poses and the position of 3D points

[1] Chen, Yu, Yisong Chen, and Guoping Wang. "Bundle adjustment revisited." arXiv preprint arXiv:1912.03858 (2019).

ORB-SLAM : Feature Based Visual SLAM

IEEE TRANSACTIONS ON ROBOTICS

ORB-SLAM: a Versatile and Accurate Monocular SLAM System

Raúl Mur-Artal*, J. M. M. Montiel, *Member, IEEE,* and Juan D. Tardós, *Member, IEEE,*

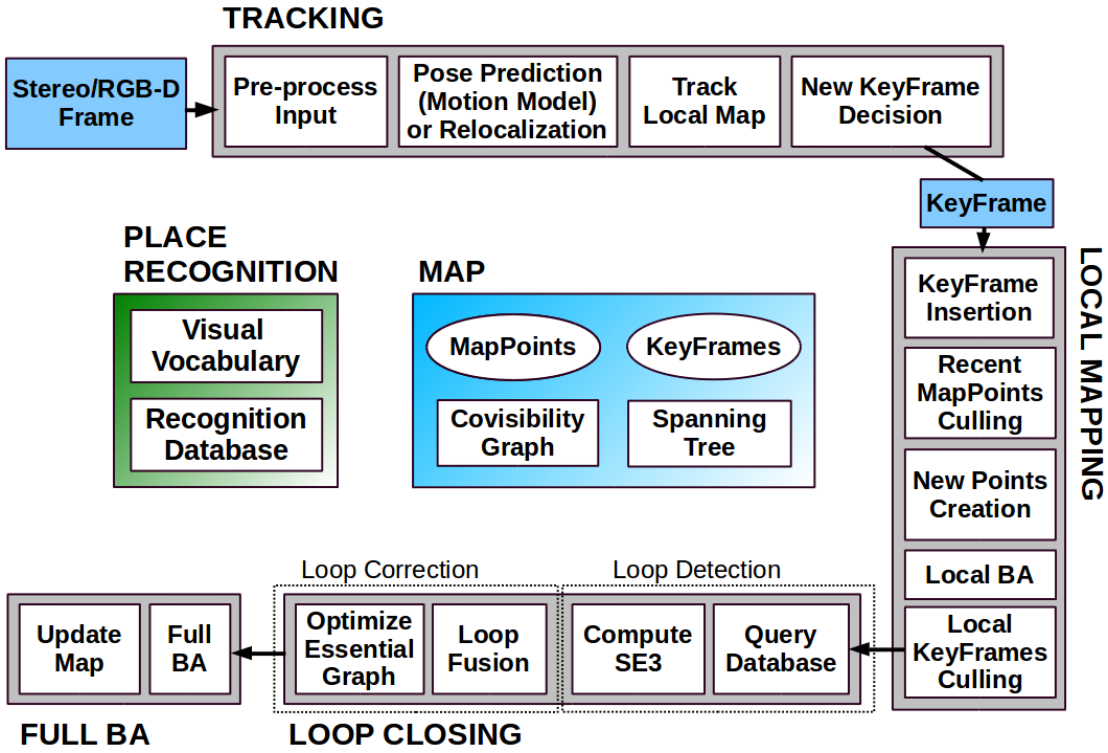
ORB-SLAM2: an Open-Source SLAM System for Monocular, Stereo and RGB-D Cameras

Raúl Mur-Artal and Juan D. Tardós

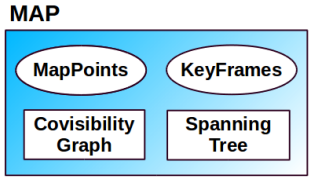
[1] Mur-Artal, Raul, Jose Maria Martinez Montiel, and Juan D. Tardos. "ORB-SLAM: a versatile and accurate monocular SLAM system." IEEE transactions on robotics 31.5 (2015)
[2] Mur-Artal, Raul, and Juan D. Tardós. "Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras." IEEE transactions on robotics 33.5 (2017)

ORB-SLAM2 Overview

- Three threads running in parallel
  - Tracking
  - Local Mapping
  - Loop Closing
- Map module
- Place recognition module



ORB-SLAM2 Map Module



Point
- 3D locations
- Viewing direction
- ORB descriptor
- Viewing distance

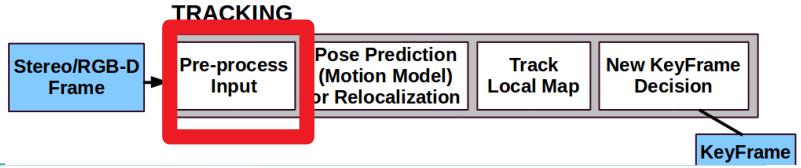
Keyframes
- Camera pose
- Camera intrinsics
- ORB features

Covisibility Graph[1]
- Nodes : Keyframes
- Edge : share observation of same map points (min 15)

Essential Graph
- Subgraph of covisibility graph
- High covisibility (min 100), spanning tree, loop closure edges

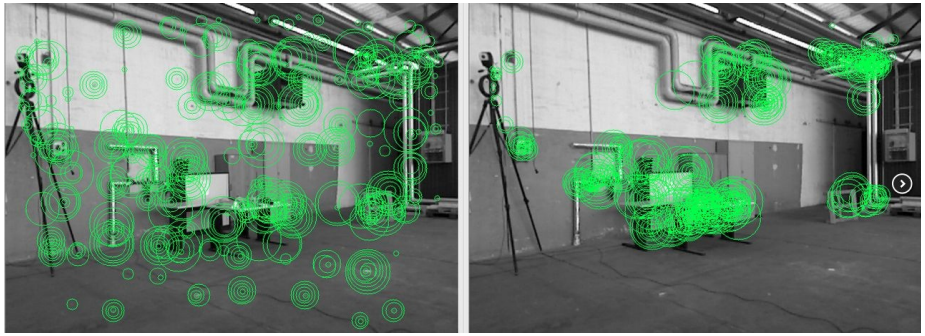
[1]. Strasdat, Hauke, et al. "Double window optimisation for constant time visual SLAM." 2011 international conference on computer vision. IEEE, 2011.

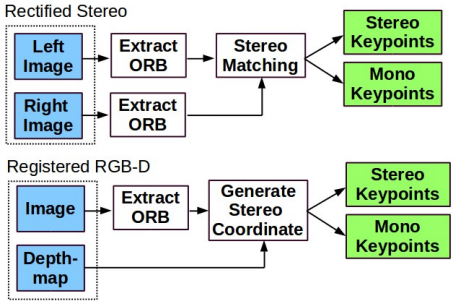
ORB-SLAM2 : Tracking Thread



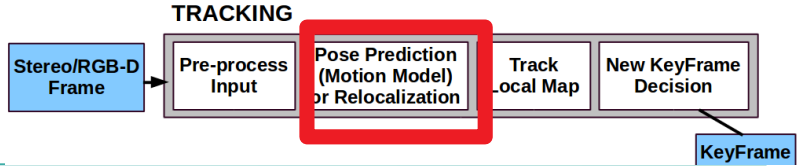
Preprocess input

- Extract ORB Features at Keypoint locations (FAST Corner Detector)





ORB-SLAM2 : Tracking Thread



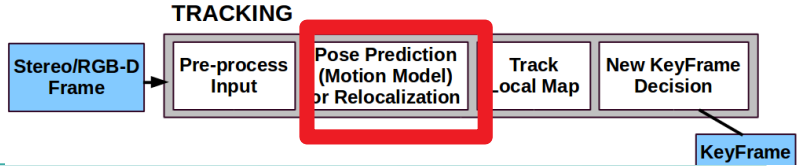
Pose Estimation from Previous Frame

- Constant velocity model assumption to predict the camera pose
- Guided search of projected map points
- Pose Optimization

Pose Estimation via Relocalization (If tracking is lost)

- Create bag of words from the frame
- Query the place recognition database: Get matching keyframes
- Outlier Rejection: RANSAC
- Get pose using PnP
- Guided search of projected map points
- Pose Optimization

ORB-SLAM2 : Tracking Thread

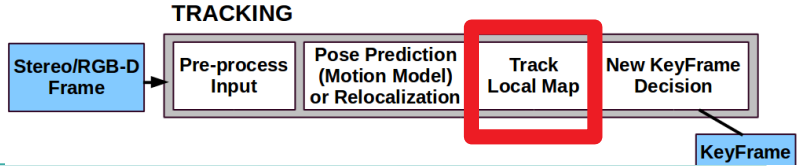


Pose Optimization

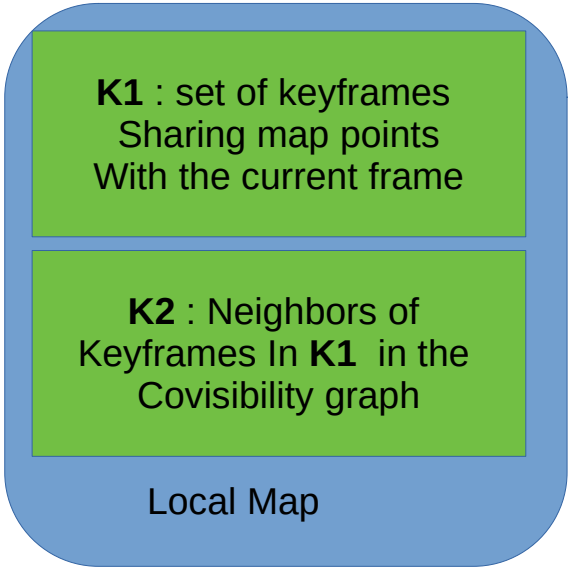
- Optimize the camera orientation R and translation t
- Minimize the error between the matched 3D points in world coordinates and key points

$$\{\mathbf{R},\mathbf{t}\} = \underset{\mathbf{R},\mathbf{t}}{\operatorname{argmin}} \sum_{i\in\mathcal{X}} \rho\left(\left\|\mathbf{x}^i_{(\cdot)} - \pi_{(\cdot)}\left(\mathbf{R}\mathbf{X}^i + \mathbf{t}\right)\right\|^2_{\Sigma}\right)$$

ORB-SLAM2 : Tracking Thread

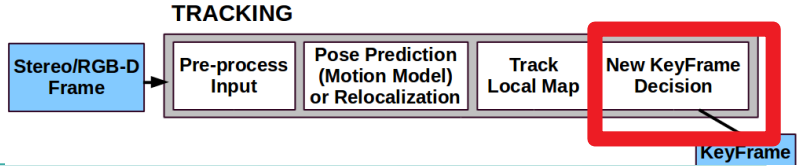


- Search for more map point correspondences in the local map
- Again do pose optimization



- **Track Local Map**
  - Project in the current frame
    - X (outside frame?) discard
  - Angle between current viewing direction and map point mean viewing direction
    - Less than threshold? discard
  - Distance is within the min and max. No? Discard
  - Compare map point descriptor with unmatched ORD features in the frame near X, take best match

ORB-SLAM2 : Tracking Thread

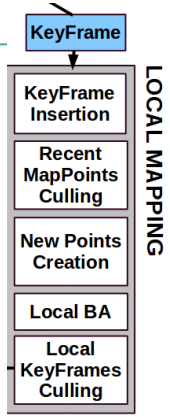


Insert new keyframes (if all criterias satisfied)

- More than 20 frames must have passed fron the last global relocalization
- Local mapping is idle, or more than 20 frames have passed from the last keyframe insertion.
- Current frame tracks at least 50 points
- Current frame tracks less than 90% points than reference keyframe

ORB-SLAM2 : Local mapping

Process new keyframes



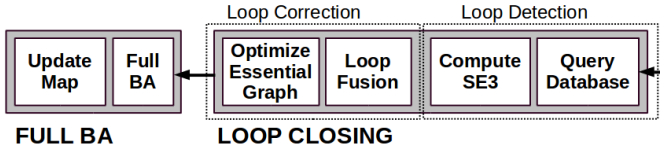
Keyframe Insertion
- Update the covisibility graph
  - Add new node and update edges
- Update spanning tree in essential graph
  - Link frame with shared points
- Comput BOW representations

Recent MapPoints Culling
- Remove points which are already visible in other keyframes
- Visible in atleast three keyframes

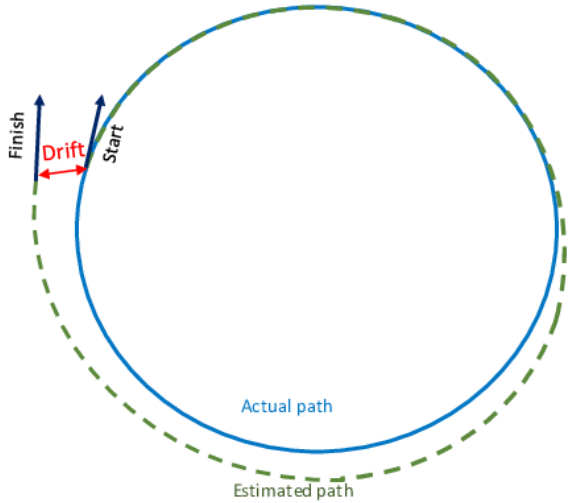
New Map Point Creation
- For unmatched keypoints in new Keyframe search match in the Connected keyframes in covisibility Graph
- Traingulate ORB keypoint pairs

Keyframe Culling
- Remove keyframe from the connected Keyframes whose 90% points cane be Seen in at least three other keyframes

ORB-SLAM2 : Loop Closing & Full BA



Loop Detection
- Find keyframes with less similarity in the covisibility graph
- Loop candidates
- Find the best candidate using key point matching
- Find similarity transform SE(3)



Loop Correction
- Update covisibility graph
- Correct keyframe poses

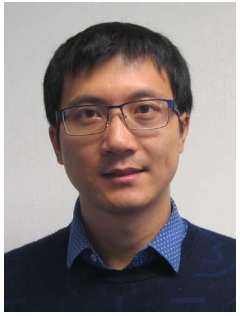
Full Bundle Adjustment
- Optimize all keyframes and points in the map

Self-Improving Geometric-CNN Framework For 3D Perception

ECCV'20 ONLINE
23-28 AUGUST 2020

Pseudo RGB-D for Self-Improving Monocular SLAM and Depth Prediction

Lokender Tiwari



Pan Ji

Quoc-Huy Tran

Bingbing Zhuang

Saket Anand

Manmohan Chandraker

Project Page: https://lokender.github.io/projects/pseudo-rgbd-slam/

US Patent : 11468585

INDRAPRASTHA INSTITUTE of INFORMATION TECHNOLOGY DELHI

NEC LABORATORIES AMERICA, INC.
Relentless passion for innovation

UC San Diego

Motivation

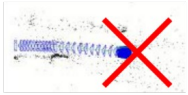
- Simultaneous Localization and Mapping (SLAM)
  - Method to construct or update a map of an unknown environment while simultaneously localizing the camera location within it

- Feature based Monocular RGB SLAM
  - Monocular = Single RGB Camera

- Challenging scenarios
  - forward only motion
  - map creation require triangulation step
    - require sufficient parallax

Geometric Monocular **RGB** SLAM

(a)



(b)

Fig . (a) RGB ORB-SLAM2 [1] fails on KITTI [2] Odometry Sequence 01, (b) Sample Image of the sequence

- Drift is a major known issue
- Localization error accumulates over time -> Drift

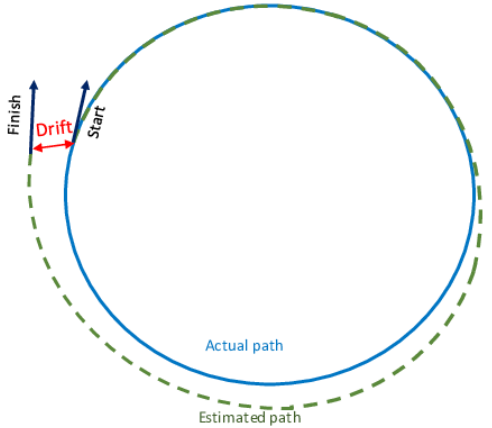
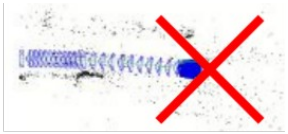


Fig . An example of drift [2]

[1] Mur-Artal et. al. ORB-SLAM2: An open-source slam system for monocular, stereo, and rgb-d cameras, IEEE Transactions on Robotics, 2017
[2] Geiger et.al., The KITTI Vision Benchmark Suite, CVPR, 2012

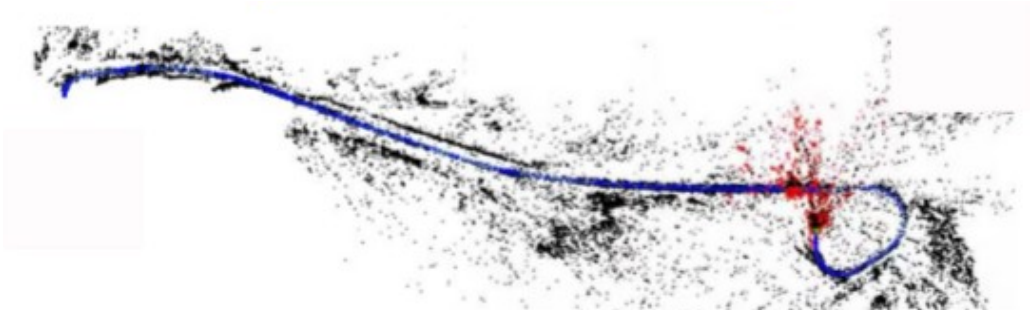
Motivation

- RGB-D SLAM:
  - less drift (better camera localization)
  - Depth from active depth sensor



Tracking Failure

(a) Monocular RGB SLAM [1]



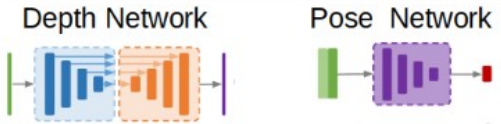
(a) RGB-D SLAM

Fig . Qualitative result. Monocular RGB vs RGB-D SLAM on KITTI [2] odometry Sequence 01,

[1] Mur-Artal et. al. ORB-SLAM2: An open-source slam system for monocular, stereo, and rgb-d cameras, IEEE Transactions on Robotics, 2017
[2] Geiger et.al., The KITTI Vision Benchmark Suite, CVPR, 2012

Motivation

- Self-supervised depth estimation model [1][2]



MonoDepth2 [1]

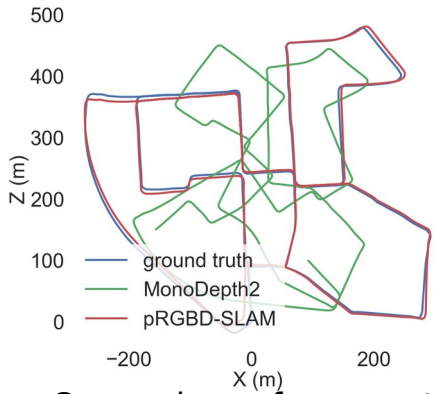
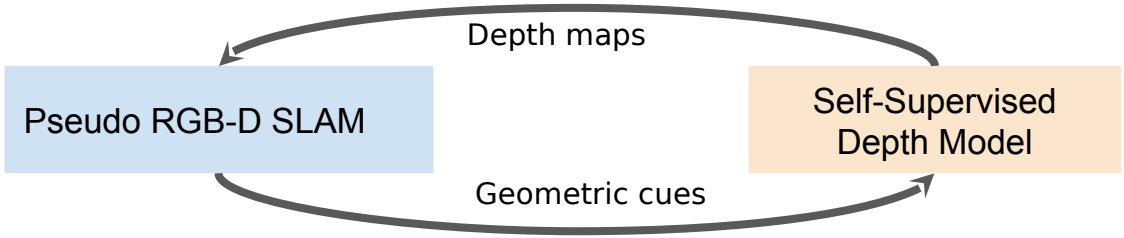


Fig . Comparison of camera trajectories

Fig . Monodepth2 estimation errors

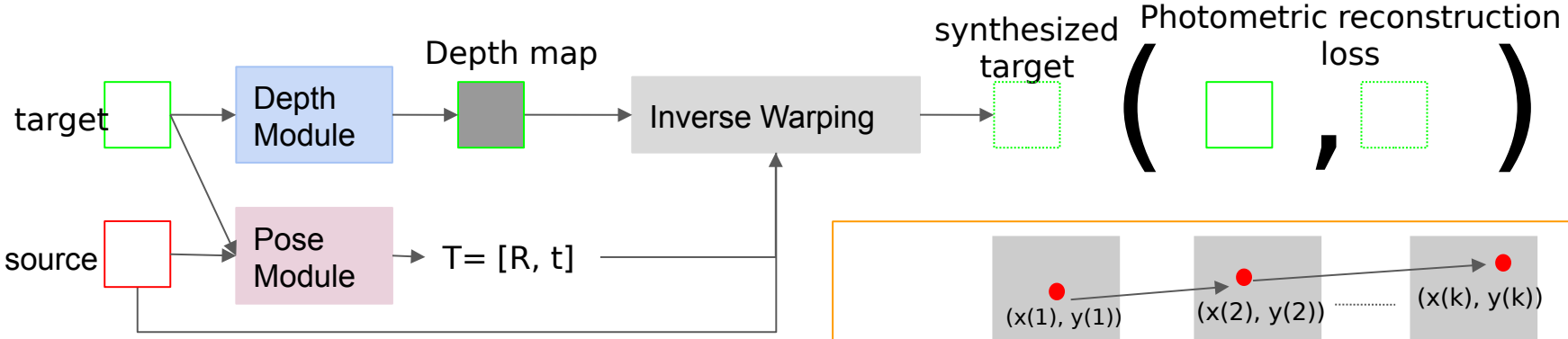


Self-Improving Geometric-CNN Framework

[1] Godard et.al, Digging into Self-Supervised Monocular Depth Prediction, ICCV, 2019
[2] Zhou et al., “Unsupervised Learning of Depth and Ego-Motion from Video”, CVPR 2017

Introduction

- Self-supervised depth estimation models [1][2]
  - view synthesis based approach
  - photometric supervision



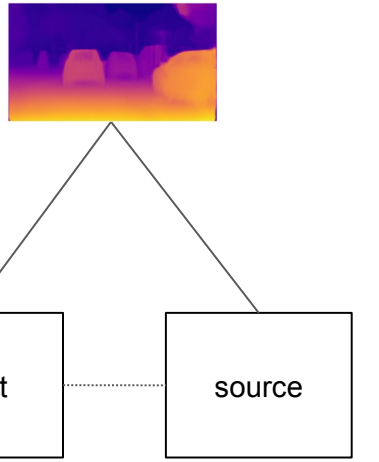
- Brightness constancy assumption
  - limitation ?

- Scene point moving through an image sequence
- Assumption: Brightness of the point will remain the same

$$I(x(t), y(t), t) = C$$

[1] Godard et.al, Digging into Self-Supervised Monocular Depth Prediction, ICCV, 2019
[2] Zhou et al., "Unsupervised Learning of Depth and Ego-Motion from Video", CVPR 2017

Introduction



- Frame gap: < 5 (Narrow baseline)
- Photometric Constraint

- Reliable depth estimates for distant scene point require wider baseline [1]

[1] Hartley and Zisserman. "Multiple View Geometry in Computer Vision, Cambridge University Press, 2001

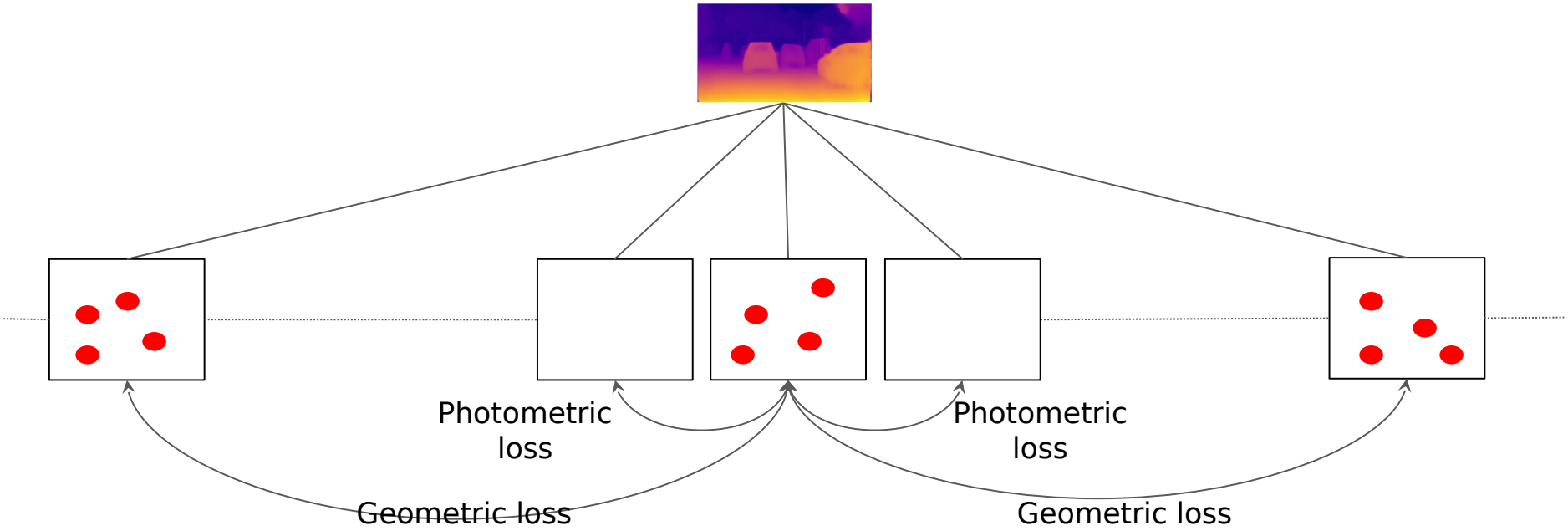
Introduction

target

source

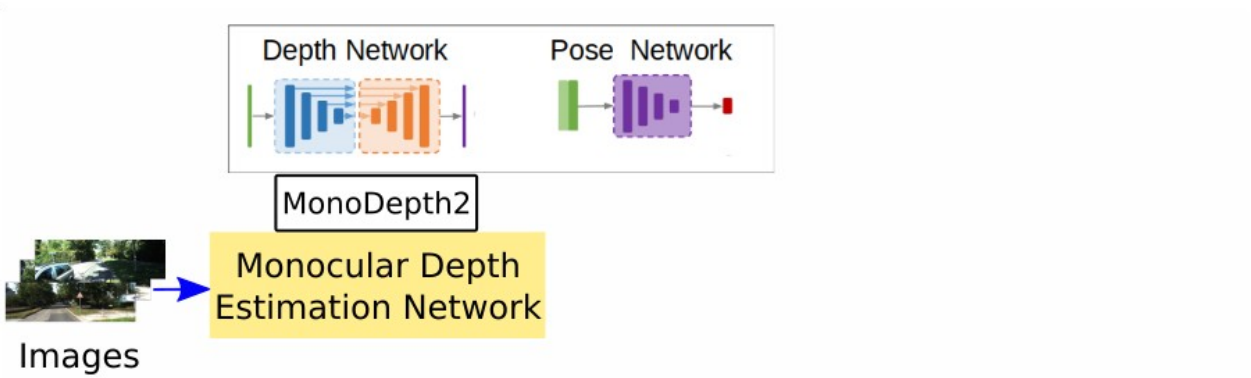
- Frame gap: relatively large (Wide baseline)
- Photometric loss become *Ineffective*
  - violation of brightness constancy assumption
  - outliers: occluded regions due to dynamic objects or change in viewpoint

Self-Improving Geometric-CNN Framework



- Combined narrow and wide baseline constraints
- Dense photometric constraint (narrow baseline)
- Geometric constraint (wide baseline)
  - on commonly visible 2D keypoints
  - Invariant to illumination and viewpoint change

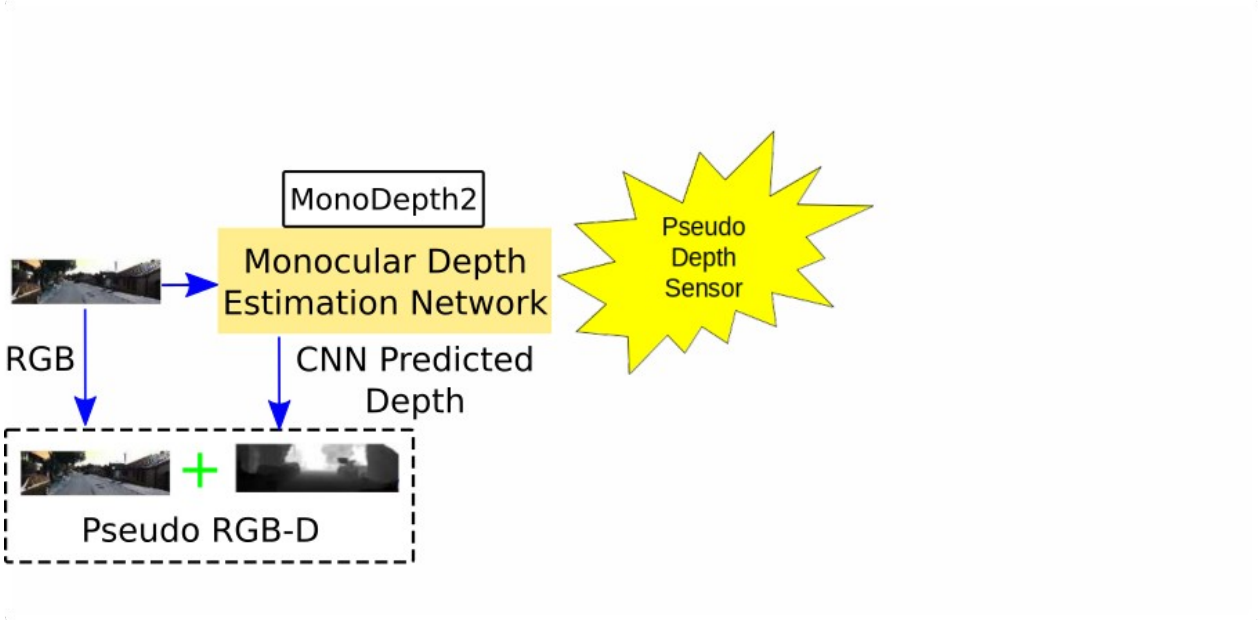
Self-Improving Geometric-CNN Framework



- Train MonoDepth2[1] using monocular image sequences from KITTI [2] dataset

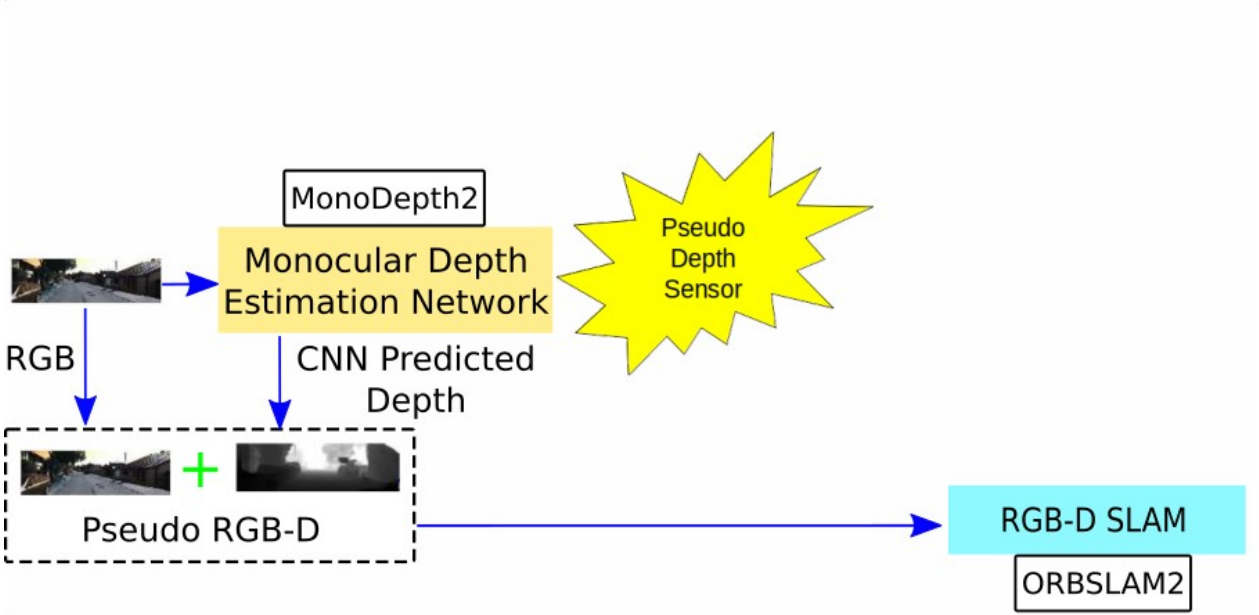
[1] Godard et.al, Digging into Self-Supervised Monocular Depth Prediction, ICCV, 2019
[2] Geiger et.al., The KITTI Vision Benchmark Suite, CVPR, 2012

Self-Improving Geometric-CNN Framework



- Prepare Pseudo RGB-D data

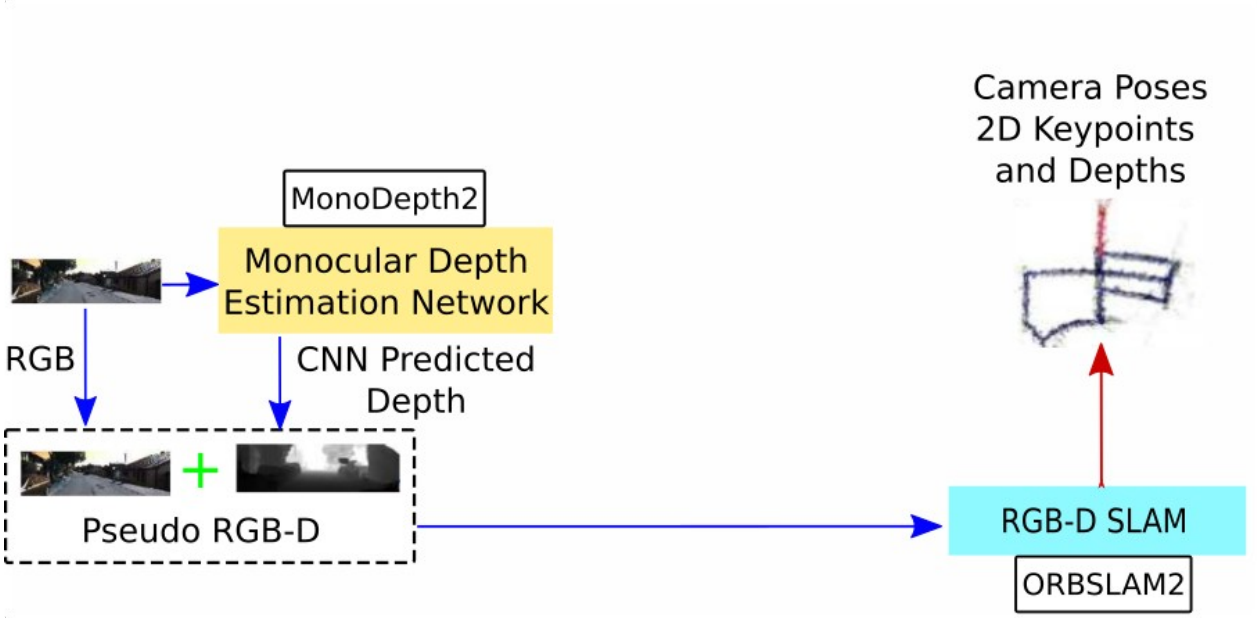
Self-Improving Geometric-CNN Framework



- Prepare Pseudo RGB-D data
- Run RGB-D SLAM [1] on Pseudo RGB-D data

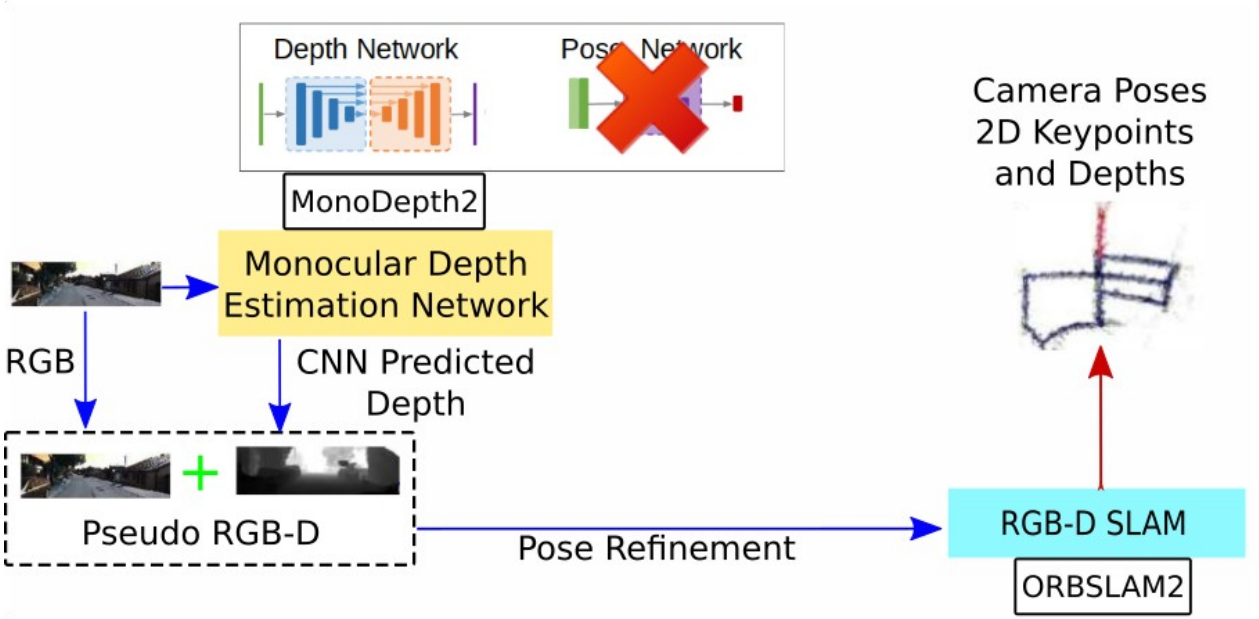
[1] Mur-Artal et. al. ORBSLAM2: An open-source slam system for monocular, stereo, and rgb-d cameras, IEEE Transactions on Robotics, 2017

Self-Improving Geometric-CNN Framework



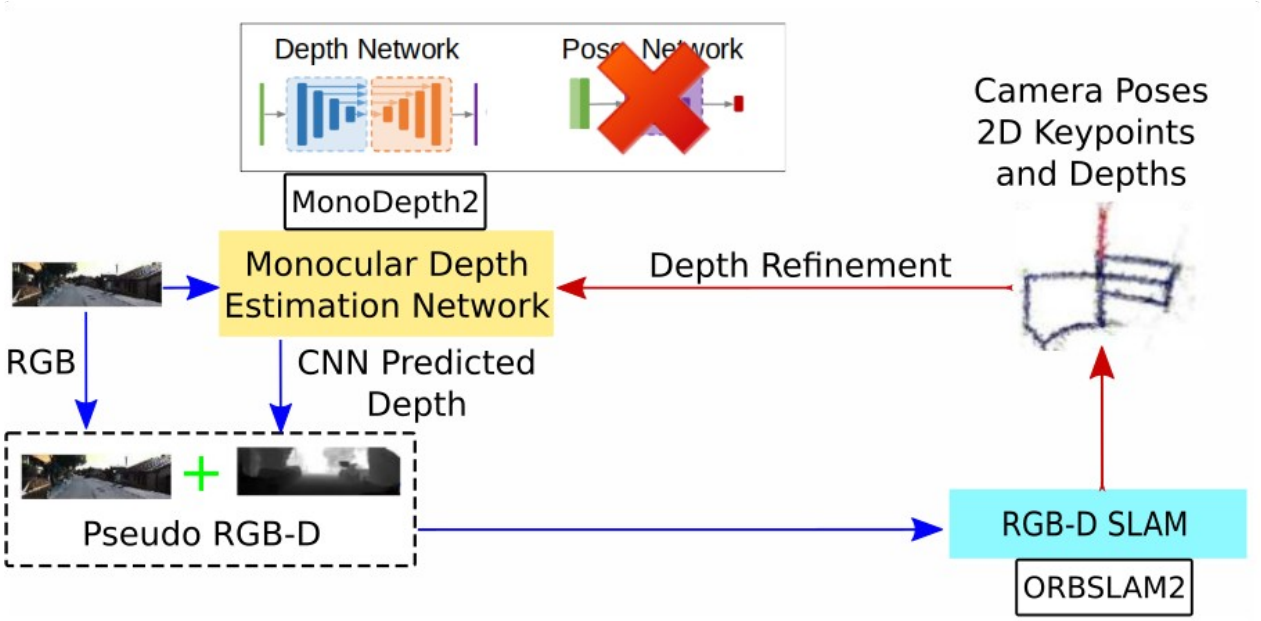
- Get Pseudo RGB-D SLAM outputs
  - Camera poses
  - tracked 2D keypoints and their depth values

Self-Improving Geometric-CNN Framework



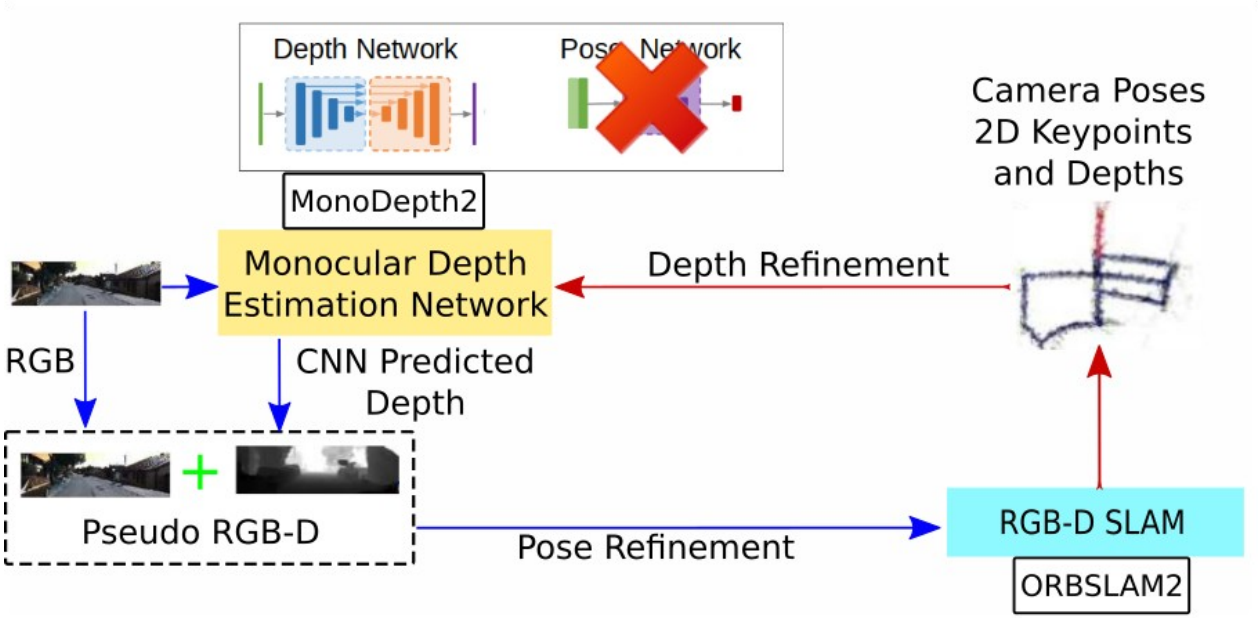
- Depth Refinement: Improve depth model
  - Disable MonoDepth2 pose network
  - Use camera poses obtained from Pseudo RGB-D SLAM

Self-Improving Geometric-CNN Framework



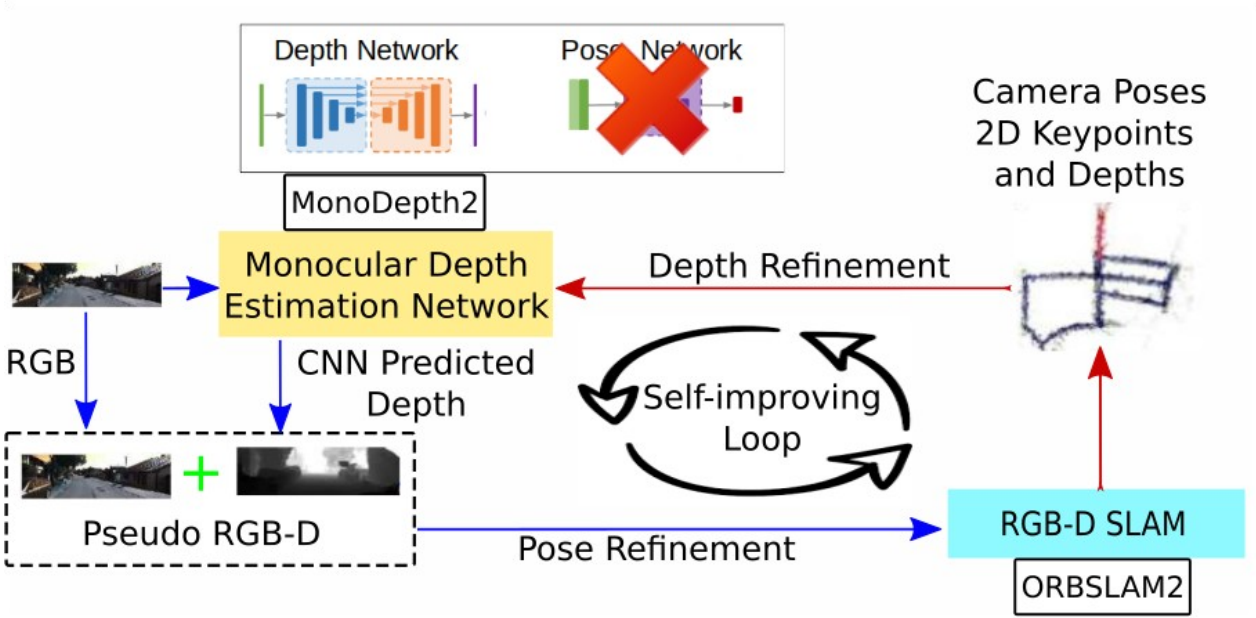
- Depth Refinement: Improve depth model
  - Disable MonoDepth2 pose network
  - Use camera poses obtained from Pseudo RGB-D SLAM

Self-Improving Geometric-CNN Framework



- Pose Refinement: Improve the camera pose estimate of the SLAM
  - Use the refined depth model to prepare Pseudo RGB-D data
  - Re-run Pseudo RGBD-D SLAM
  - Get refined camera poses, keypoints and their updated locations

Self-Improving Geometric-CNN Framework



- Self-Improving Loop
  - Run until we see no improvement in depth or pose

Self-Improving Geometric-CNN Framework

- **Depth Refinement Losses**

$$\mathcal{L} = \alpha\mathcal{P} + \mu(\mathcal{T}_{c\leftrightarrow k1} + \mathcal{T}_{c\leftrightarrow k2} + \mathcal{T}_{k1\leftrightarrow k2}) + \gamma\mathcal{D}_c + \beta\mathcal{S}_c$$

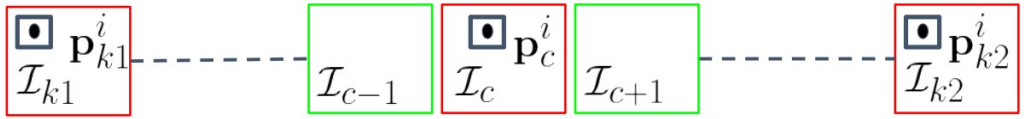
Photometric Reconstruction Loss (narrow baseline)

Symmetric Depth Transfer Losses (Wide baseline)

Depth Consistency Loss

Smoothness Loss

Input frames



$$\mathcal{P} = PE(\mathcal{I}'_{c-1}, \mathcal{I}_{c-1}) + PE(\mathcal{I}'_{c+1}, \mathcal{I}_{c+1}) \quad (1)$$

$$\mathcal{T}_{c\leftrightarrow k1}(\mathbf{w}) = |d_{c\to k1}(\mathbf{w}) - d_{k1}(\mathbf{w})| + |d_{k1\to c}(\mathbf{w}) - d_c(\mathbf{w})| \quad (2)$$

$$\mathcal{D}_c = \sum_i |d^i_c(\mathbf{w}) - d^i_c(\mathbf{SLAM})| \quad (3)$$

$$\mathcal{S}_c = |\partial_x d_c|\, e^{-|\partial_x I_t|} + |\partial_y d_c|\, e^{-|\partial_y I_t|} \quad (4)$$

Experiments

- KITTI dataset [1]
- **Error Metrics**

$\tilde{d}_i$: ground truth depth

$d_i^\star$: estimated depth

Absolute Relative

$$\frac{1}{K}\sum_{i=1}^{K}\frac{|\tilde{d}_i - d_i^\star|}{d_i^\star}$$

Squared Relative

$$\frac{1}{K}\sum_{i=1}^{K}\frac{||\tilde{d}_i - d_i^\star||^2}{d_i^\star}$$

RMSE

$$\sqrt{\frac{1}{K}\sum_{i=1}^{K}(\tilde{d}_i - d_i^\star)^2}$$

RMSE Log

$$\sqrt{\frac{1}{K}\sum_{i=1}^{K}\| \log_{10}(\tilde{d}_i) - \log_{10}(d_i^\star)\|^2}$$

- **Accuracy Metrics**
  - a1, a2, a3 $\max(\frac{d_i^\star}{\tilde{d}_i}, \frac{\tilde{d}_i}{d_i^\star}) = \delta < t \ (t \in [1.25, 1.25^2, 1.25^3])$

Method	Train	Lower is better				Higher is better		
		Abs Rel	Sq Rel	RMSE	RMSE log	a1	a2	a3
Yang [2]	M	0.182	1.481	6.501	0.267	0.725	0.906	0.963
Mahjourian [3]	M	0.163	1.240	6.220	0.250	0.762	0.916	0.968
DDVO [4]	M	0.151	1.257	5.583	0.228	0.810	0.936	0.974
Ranjan [5]	M	0.148	1.149	5.464	0.226	0.815	0.935	0.973
WBAF [6]	M	0.135	0.992	5.288	0.211	0.831	0.942	0.976
MonoDepth2 [7]	M	0.117	0.941	4.889	0.194	0.873	0.957	0.980
pRGBD-Refined (Proposed)	M	**0.113**	**0.793**	**4.655**	**0.188**	**0.874**	**0.960**	**0.983**

Tab 1. Quantitative depth evaluation results

[1] Geiger et al. “The KITTI Vision Benchmark Suite”, CVPR, 2012
[2] Yang, et al. “Unsupervised learning of geometry with edge-aware depth-normal consistency. AAAI, 2018
[3] Mahjourian, et al. “Unsupervised learning of depth and ego-motion from monocular video using 3d geometric constraints. CVPR, 2018
[4] Wang, et al. “Learning depth from monocular videos using direct methods. CVPR, 2018
[5] Ranjan, et al. “Competitive collaboration: Joint unsupervised learning of depth, camera motion, optical flow and motion segmentation. CVPR, 2019
[6] Zhou, et al. “Windowed bundle adjustment framework for unsupervised learning of monocular depth estimation with u-net extension and clip loss. IEEE RAL, 2020
[7] Godard et.al, Digging into Self-Supervised Monocular Depth Prediction, ICCV, 2019

Experiments

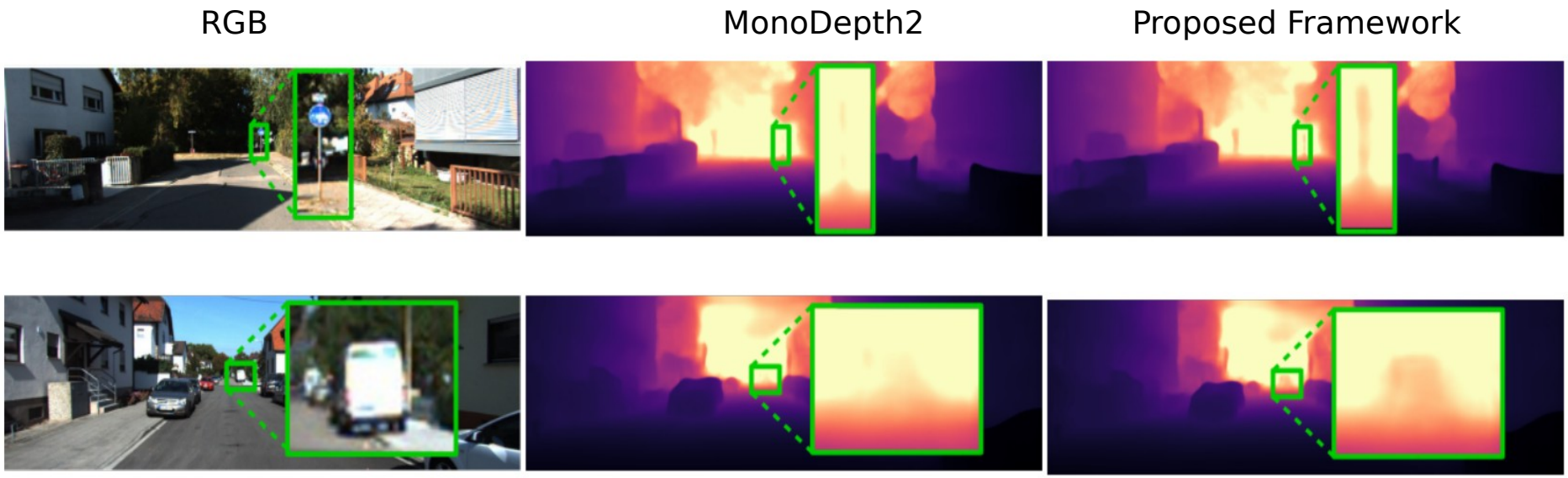


Fig . Qualitative depth evaluation

Experiments

- KITTI dataset [1]
  - Test set (sequence, 09 and 10)
- **Error Metrics**
  - RMSE
  - Relative Translation
  - Relative Rotation
- Supervision
  - GT-Sup: Ground truth supervision
  - Self-Sup: Self supervision

Method	Train	Seq. 09			Seq. 10		
		RMSE	Rel Tr	Rel Rot	RMSE	Rel Tr	Rel Rot
DeepV2D [2]	GT-Sup	79.06	8.71	0.037	48.49	12.81	0.083
SfMLearner [3]	Self-Sup	24.31	8.28	0.031	20.87	12.20	0.030
CC [4]	Self-Sup	29.00	6.92	0.018	13.77	7.97	0.031
DeepMatchVO [5]	Self-Sup	27.08	9.91	0.038	24.44	12.18	0.059
Monodepth2 [6]	Self-Sup	55.47	11.47	0.032	20.46	7.73	0.034
RGB ORB-SLAM2 [7]	–	18.34	7.42	**0.004**	8.90	5.85	**0.004**
pRGBD-SLAM	Self-Sup	**11.97**	**4.20**	0.010	**6.35**	**4.40**	0.016

Tab 2. Quantitative pose evaluation results

[1] Geiger et al. “The KITTI Vision Benchmark Suite”, CVPR, 2012
[2] Teed, et al. “DeepV2D: Video to depth with differentiable structure from motion. arXiv preprint arXiv:1812.04605 (2018)
[3] Zhou, et al. “Unsupervised learning of depth and ego-motion from video. In: CVPR. pp. 1851–1858 (2017)
[4] Ranjan et al. “ Competitive collaboration: Joint unsupervised learning of depth, camera motion, optical flow and motion segmentation. In: CVPR (2019)
[5] Shen, et al. “Beyond photometric loss for self-supervised ego-motion estimation. In: ICRA (2019)
[6] Godard et.al, Digging into Self-Supervised Monocular Depth Prediction, ICCV, 2019
[7] Mur-Artal et. al. ORBSLAM2: An open-source slam system for monocular, stereo, and rgb-d cameras, IEEE Transactions on Robotics, 2017

Analysis of Self-Improving Loops

- Self-improving loops
  - error plots (b) (c) and (d)
  - accuracy plot (a)
  - computed at different depth threshold

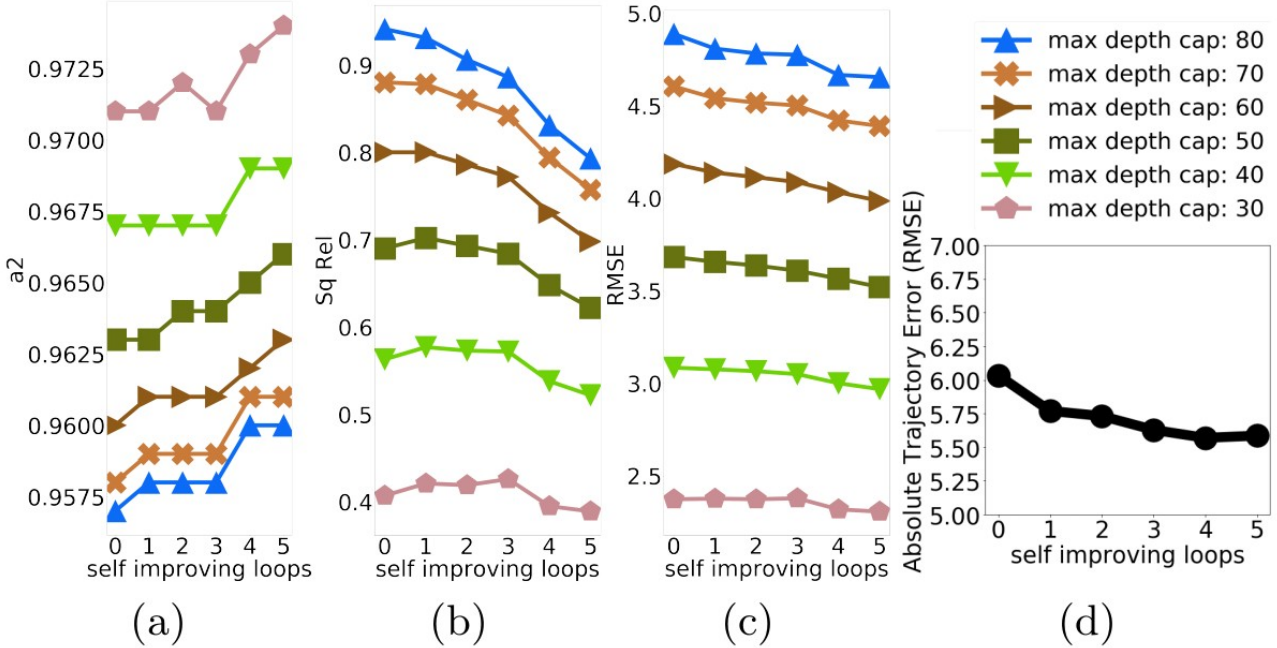


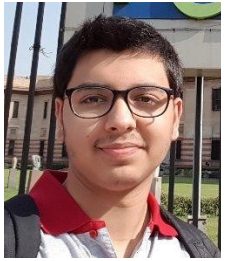
Fig 11. Analysis of self-improving loops

Robust Image Classification

WACV
WAIKOLOA, HI JAN 4-8

REGroup: Rank-aggregating Ensemble of Generative Classifiers for Robust Predictions

Lokender Tiwari



Anish Madan

Saket Anand

Subhashis Banerjee

Project Page: https://lokender.github.io/projects/REGroup/

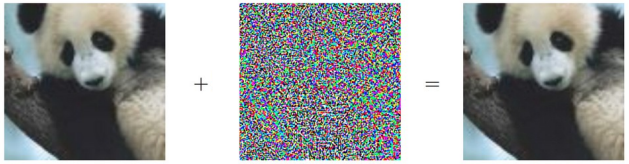
INDRAPRASTHA INSTITUTE of INFORMATION TECHNOLOGY DELHI

tcs Research

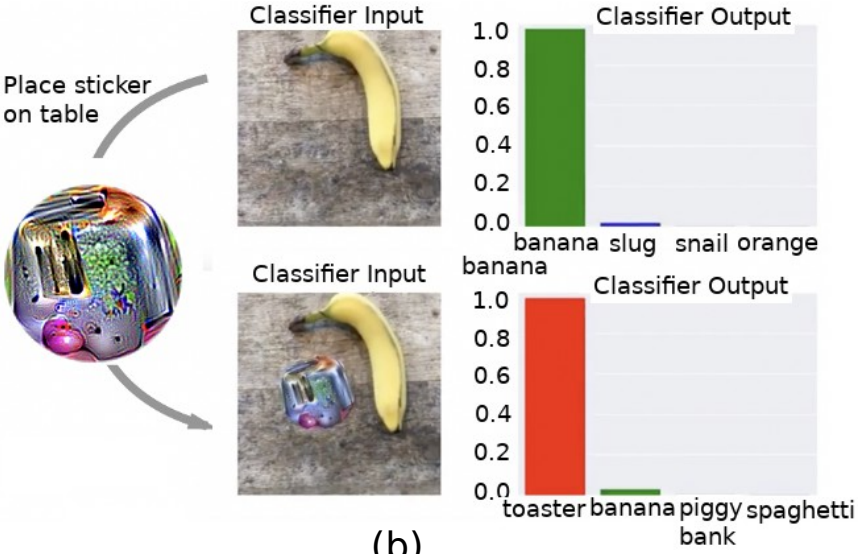
भारतीय प्रौद्योगिकी संस्थान दिल्ली
INDIAN INSTITUTE OF TECHNOLOGY DELHI

Motivation

- Deep Neural Network based image classifiers can be fooled by adversarial examples
- Clean sample: X
- Adversarial sample: X + δ
- Adversarial example generation methods
  - find minimum perturbation(δ)



(a)



(b)

Fig . Adversarial Examples: (a) example from [2], (b) Adversarial Patch [3]

[1] Szegedy et al. “Intriguing properties of neural networks”. CoRR, 2013
[2] Goodfellow et al. “Explaining and Harnessing Adversarial Examples”, ICLR, 2015
[2] Brown et al. “Adversarial Patch”, NIPS Workshop, 2017
[3] Yao et al. “Describing the Scene as a Whole: Joint Object Detection, Scene Classification and Semantic Segmentation”, CVPR, 2012.

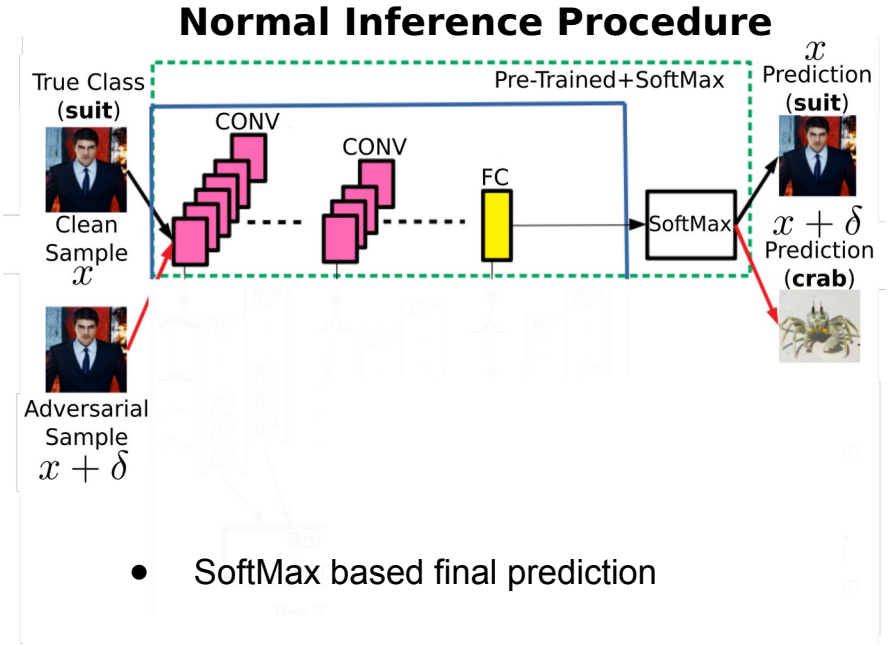
Motivation

- Successful defenses:
  - Adversarial Training [1] : Train classifier using both clean and adversarial samples
  - Input randomization [2] before passing to a classifier
- Require fine-tuning or retraining (computational expensive and time consuming)
  - Adversarial training for full scale ImageNet classification
    - 52 hours on 128 NVIDIA V100 GPUs for ResNet-152 based classifier model [1]
- Most defense methods [3]
  - are attack specific, architecture specific
  - practically not scalable (e.g., full ImageNet level)
- Need for a defense mechanism
  - agnostic to classifier architectures and the adversarial attack generation method
  - can detect and make correct prediction for adversarial examples
  - easy to scale to large scale classification task

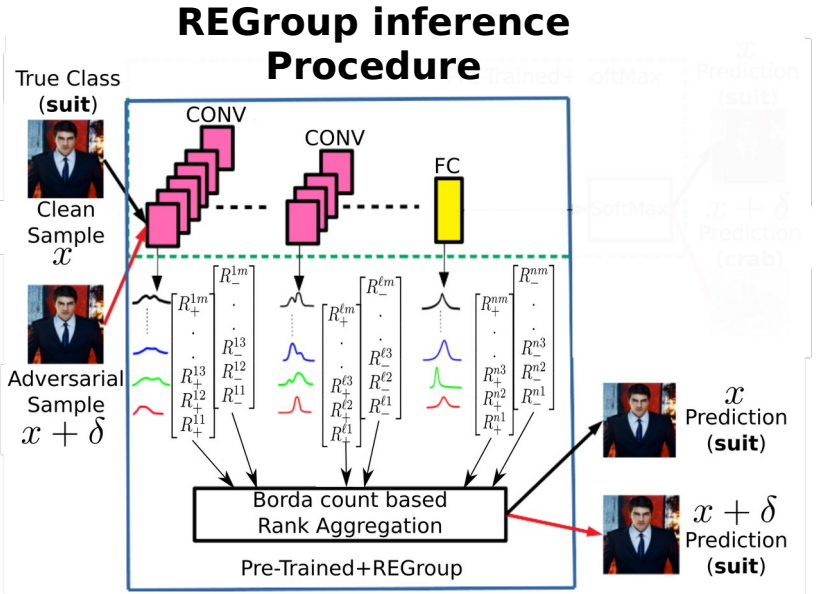
- **REGroup: Rank-aggregating Ensemble of Generative classifiers for robust predictions**

[1] Cihang Xie et al. “Feature Denoising for Improving Adversarial Robustness”. CVPR, 2019
[2] Edward Raff et al. “Barrage of Random Transforms for Adversarially Robust Defense”. CVPR, 2019
[3] Madry et al, https://www.robust-ml.org/

REGroup Overview



- SoftMax based final prediction



- Layer-wise ranked predictions
- Final prediction based on aggregated rankings

REGroup Overview

1. Generative classifiers
    - Class conditional layer-wise mixture distributions
        - Two distributions per layer, per class
            - Using positive pre-activation neural responses
            - Using negative pre-activation neural responses

] One time only

2. Inference step
    - Layer-wise comparison of sample distributions with the class conditional distributions
        - KL-Divergence
    - Make layer-wise ranked predictions
    - Use robust rank aggregation strategy to aggregate ranked predictions
    - Final prediction is the class, with the highest rank

Class conditional layer-wise generative classifiers

- Layerwise neural response distributions

  Rectified Linear Unit (ReLU): $relu(x) = \max(0, x)$

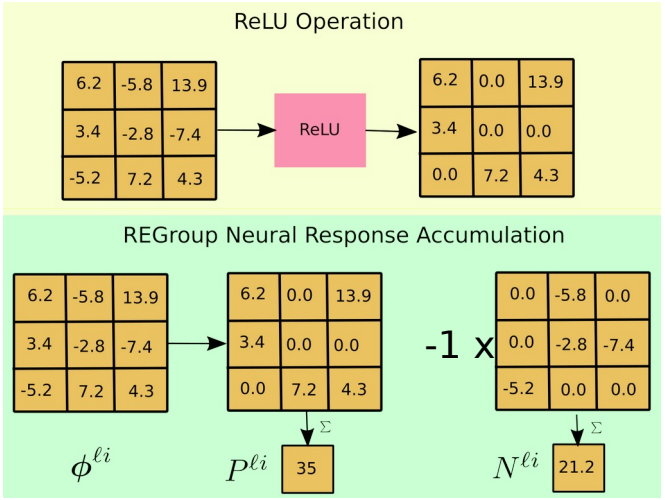
  - Truncate negative pre-activation during forward pass
  - we found negative pre-activations also contains semantically meaningful information

  Positive ($P^{\ell i}$) and Negative ($N^{\ell i}$) response accumulator

- Overall strength of positive and negative pre-activation neural responses over spatial dimension

  $\phi^{\ell i}$ i^{th} feature map of l^{th} layer (Convolutional Layers)

  $\phi^{\ell i}$ scalar neuron response (Linear Layers)



$$P^{\ell i} = \sum \max(0, \phi^{\ell i}), \quad N^{\ell i} = \sum \max(0, -\phi^{\ell i}), \quad \forall i \qquad \textbf{(8)}$$

$$\mathbb{P}^{\ell} = \frac{P^{\ell}}{\sum_i P^{\ell i}}, \quad \mathbb{N}^{\ell} = \frac{N^{\ell i}}{\sum_i N^{\ell i}} \quad (PMFs) \qquad \textbf{(9)}$$

Class conditional layer-wise generative classifiers

- Visualization of accumulators for ResNet-50 based classifier
  - T-SNE [1] Plots for 5 random classes of ImageNet
- Pos: positive accumulator, Neg: negative accumulator
- Combined : using both positive and negative simultaneously

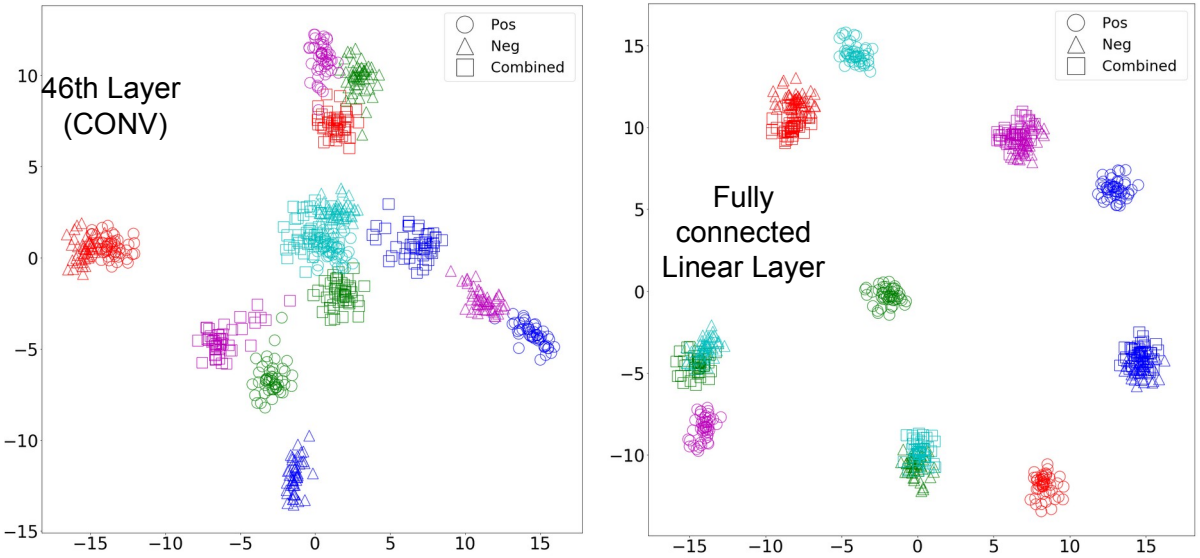


Fig . t-SNE plots of accumulated neural responses

Class conditional mixture distributions

$$\mathbb{C}_y^{+\ell} = \sum_{j:x_j \in \mathcal{S}_y} \lambda_j \mathbb{P}_j^{\ell}, \quad \mathbb{C}_y^{-\ell} = \sum_{j:x_j \in \mathcal{S}_y} \lambda_j \mathbb{N}_j^{\ell} \qquad \textbf{(10)}$$

$\mathcal{S}$: Subset of correctly classified training data

$\mathcal{S} = \{\cup_{y=1}^{M} \mathcal{S}_y\}$: $\mathcal{S}_y$ subset with label y

[1] Chan, et al. "t-SNE-CUDA: GPU-Accelerated t-SNE and its Applications to Modern Data." *International Symposium on Computer Architecture and High Performance Computing*, 2018

REGroup Inference Strategy

- $\mathbb{P}^{\ell}$ and $\mathbb{N}^{\ell}$ test sample's PMFs
- Compare $\mathbb{P}^{\ell}$ and $\mathbb{N}^{\ell}$ with the class conditional distributions $\mathbb{C}^{+\ell}$ and $\mathbb{C}^{-\ell}$ using KL-Divergence

$$P_{KL}(\ell, y) = \sum_i \mathbb{C}_y^{+\ell i} \log\left(\frac{\mathbb{C}_y^{+\ell i}}{\mathbb{P}^{\ell i}}\right), \forall y \in \{1,\ldots,M\} \qquad \textbf{(11)}$$

- Assign rank to each class based on KL-Divergence
  higher the rank, most likely the class ($R_+^{\ell y}$ is the rank of y^{th} class)

$$R_+^{\ell} = [R_+^{\ell 1}, R_+^{\ell 2}, \ldots, R_+^{\ell y}, \ldots, R_+^{\ell M}] \qquad \textbf{(12)}$$

Similarly compute $N_{KL}(\ell, y)$ and R_-^{ℓ}

Two rankings per layer

R_+^{ℓ} R_-^{ℓ}

REGroup Inference Strategy

- Rank Aggregation using Borda Count [1][2]
  - ranking based voting mechanism
  - voter ranks the list of candidates in the order of preference

$B^{\ell y}$: The Borda count for the y^{th} class at the ℓ^{th} layer

$B^{\ell y} = B^{\ell y}_{+} + B^{\ell y}_{-}$, M: total number of classes

$$B^{\ell y}_{+} = (M - R^{\ell y}_{+}), \qquad B^{\ell y}_{-} = (M - R^{\ell y}_{-}) \qquad \textbf{(13)}$$

$B^{:ky}$: Aggregated Borda counts of highest k layers

$$B^{:ky} = \sum_{\ell=n-k+1}^{n} B^{\ell y}, \quad \forall y \in \{1..M\} \qquad \textbf{(14)}$$

$$\widehat{y} = argmax_y \; B^{:ky} \qquad \textbf{(15)}$$

[1] Jörg Rothe. “Borda Count in Collective Decision Making: A Summary of Recent Results”. AAAI, 2019
[2] Black et al. “The Theory of Committees and Elections”. Cambridge University Press., 1958.

Experiments

- Classifier architectures: VGG19 [1] and ResNet50 [2]
- Dataset: ImageNet [3]
- Adversarial attacks
  - Gradient Based Attacks (White box)
    - Full access to network parameters
  - Gradient Free Attacks (Black box)
    - Classifier is a back-box

[1] Simonyan et al. “Very Deep Convolutional Networks for Large-Scale Image Recognition”, ICLR, 2015
[2] He at al. “Deep Residual Learning for Image Recognition”, CVPR, 2016
[3] Jia Deng et al. “ImageNet: A Large-Scale Hierarchical Image Database”. CVPR, 2009

Experiments

- Gradient Based Attacks:
  - Projective Gradient Descent (PGD) [1]
  - DeepFool (DFool) [2]
  - Carlini and Wagner (C&W)[3]
  - Trust Region attack (TR) [4]
  - Color adversarial attack (cAdv) [5]

UN : Untargeted Attack
TA : Targeted Attack
HC : High Confidence (at-least 90%)
ϵ : Adversarial Perturbation Budget
#S : No. of adversarial samples
T1(%) : Top-1 accuracy

	Data	UN / TA / HC	ϵ	ResNet-50 #S	ResNet-50 SMax T1(%)	ResNet-50 REGroup T1(%)	VGG-19 #S	VGG-19 SMax T1(%)	VGG-19 REGroup T1(%)
Clean	V10K	–	–	10000	100	88	10000	100	76
Clean	V2K	–	–	2000	100	86	2000	100	72
Clean	V10C	–	–	417	100	84	392	100	79
PGD	V10K	UN	4 (L_∞)	9997	0	48	9887	0	46
DFool	V10K	UN	2 (L_2)	9789	0	61	9939	0	55
C&W	V10K	UN	4 (L_2)	10000	0	40	10000	0	38
TR	V10K	UN	2 (L_∞)	10000	0	41	9103	0	45
cAdv	V10C	UN	–	417	0	37	392	0	18
PGD	V2K	TA	(L_∞)	2000	0	47	2000	0	31
C&W	V2K	TA	(L_2)	2000	0	46	2000	0	38
PGD	V2K	UN+HC	(L_∞)	2000	0	21	2000	0	19
PGD	V2K	TA+HC	(L_∞)	2000	0	23	2000	0	17

Tab 5. Classification accuracy on gradient based attacks

[1] Madry, Aleksander, et al. "Towards Deep Learning Models Resistant to Adversarial Attacks." *ICLR*. 2018
[2] Moosavi-Dezfooli et al. "Deepfool: a simple and accurate method to fool deep neural networks." *CVPR*. 2016
[3] Carlini, Nicholas, and David Wagner. "Towards evaluating the robustness of neural networks." *IEEE symposium on security and privacy*, 2017
[4] Yao, Zhewei, et al. "Trust region based adversarial attack on neural networks." *CVPR*, 2019
[5] Bhattad, Anand, et al. "Unrestricted Adversarial Examples via Semantic Manipulation." *ICLR*. 2019

Experiments

- Gradient Free Attacks (Black Box Attacks):
  - SPSA Attack [1]
  - Boundary Attack [2]
  - Spatial Attack [3]

	Data	UN / TA / HC	ϵ	ResNet-50			VGG-19		
					SMax	REGroup		SMax	REGroup
				#S	T1(%)	T1(%)	#S	T1(%)	T1(%)
SPSA	V10K	UN	4 (L_∞)	4911	0	71	5789	0	58
Boundary	V10K	UN	2 (L_2)	10000	0	50	10000	0	50
Spatial	V10K	UN	2 (L_2)	2624	0	36	2634	0	30

Tab 6. Classification accuracy on gradient free attacks

UN : Untargeted Attack
ϵ : Adversarial Perturbation Budget
#S : No. of adversarial samples
T1(%) : Top-1 accuracy

[1] Jonathan Uesato et al. “Adversarial Risk and the Dangers of Evaluating Against Weak Attacks”. ICML, 2018
[2] Brendel et al. “Decision-Based Adversarial Attacks: Reliable Attacks Against Black-Box Machine Learning Models”. ICLR, 2018
[3] Logan Engstrom et al. “Exploring the Landscape of Spatial Robustness”. ICML, 2019

Experiments

- Comparison with adversarial training [1] and input randomization method [2]
- BaRT: Barrage of Random Transforms
- EOT : Expectations over Input Transformations
- **Dataset:** Full ImageNet
- PGD Attack
  - Comparison with respect to adversarial attack strength

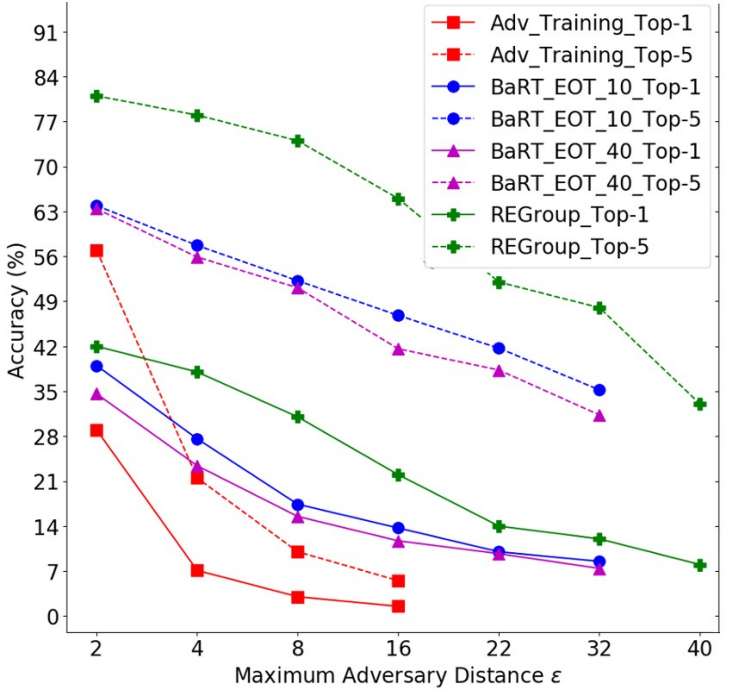


Fig 20. Comparison with adversarial training and fine tuning methods

[1] Kurakin, et al. "Adversarial machine learning at scale." ICLR, 2016
[2] Edward, et al. "Barrage of random transforms for adversarially robust defense." CVPR. 2019.

2021 ICCV OCTOBER 11-17 VIRTUAL

DeepDraper

TATA CONSULTANCY SERVICES

TCS Research & Innovation

Fast and Accurate 3D Garment Draping over a 3D Human Body

Lokender Tiwari Brojeshwar Bhowmick

TCS Research

https://www.tcs.com/tcs-research

lokender.tiwari@tcs.com

Differentiable 3D Vision and Graphics

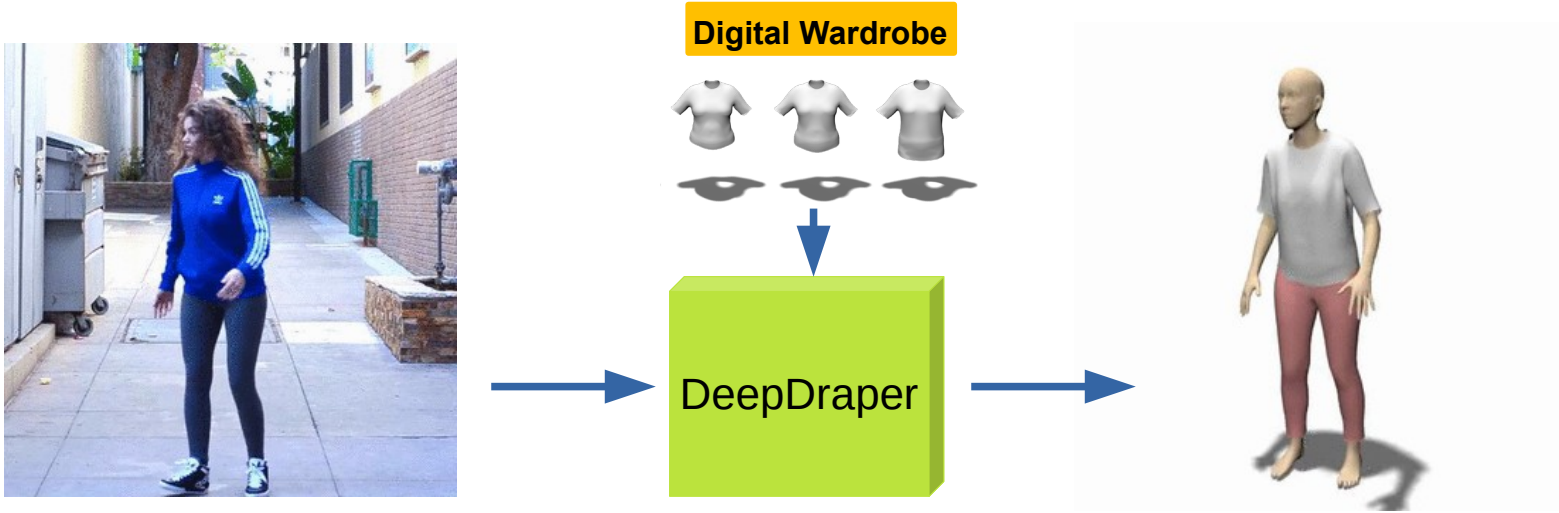
IEEE/CVF International Conference on Computer Vision, 2021

US Patent App : 17/646,330

Outline

- Motivation
- Introduction
- DeepDraper – Proposed Method
- Results
- Analysis
- Conclusion

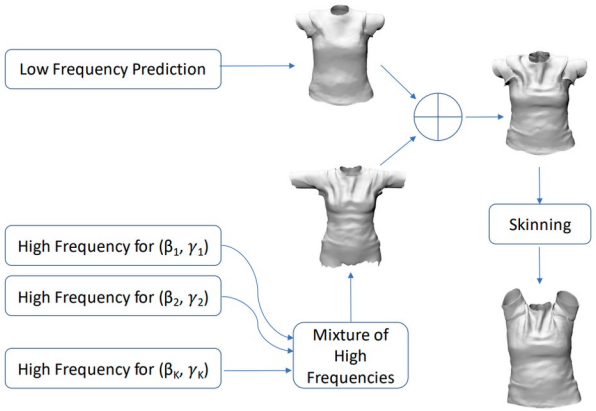
Motivation



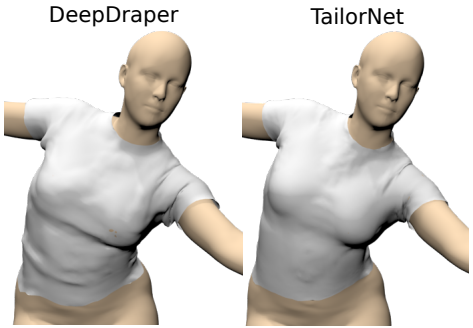
- Expectations
  - ✔ Realistic appearance
  - ✔ Accurate fitting
  - ✔ Better generalization

Introduction

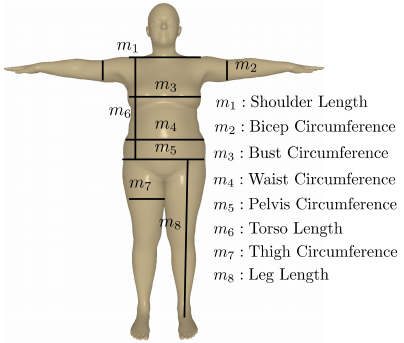
- DeepDraper
  - ✔ Unified method (e.g. mixture of fixed no. of high frequency models versus single model)
  - ✔ Garment deformations : a function of body shape, pose, measurements, and garment styles
  - ✔ Standard body measurements
  - ✔ Coupled geometric and perceptual constraints



TailorNet (CVPR 2020) (a)



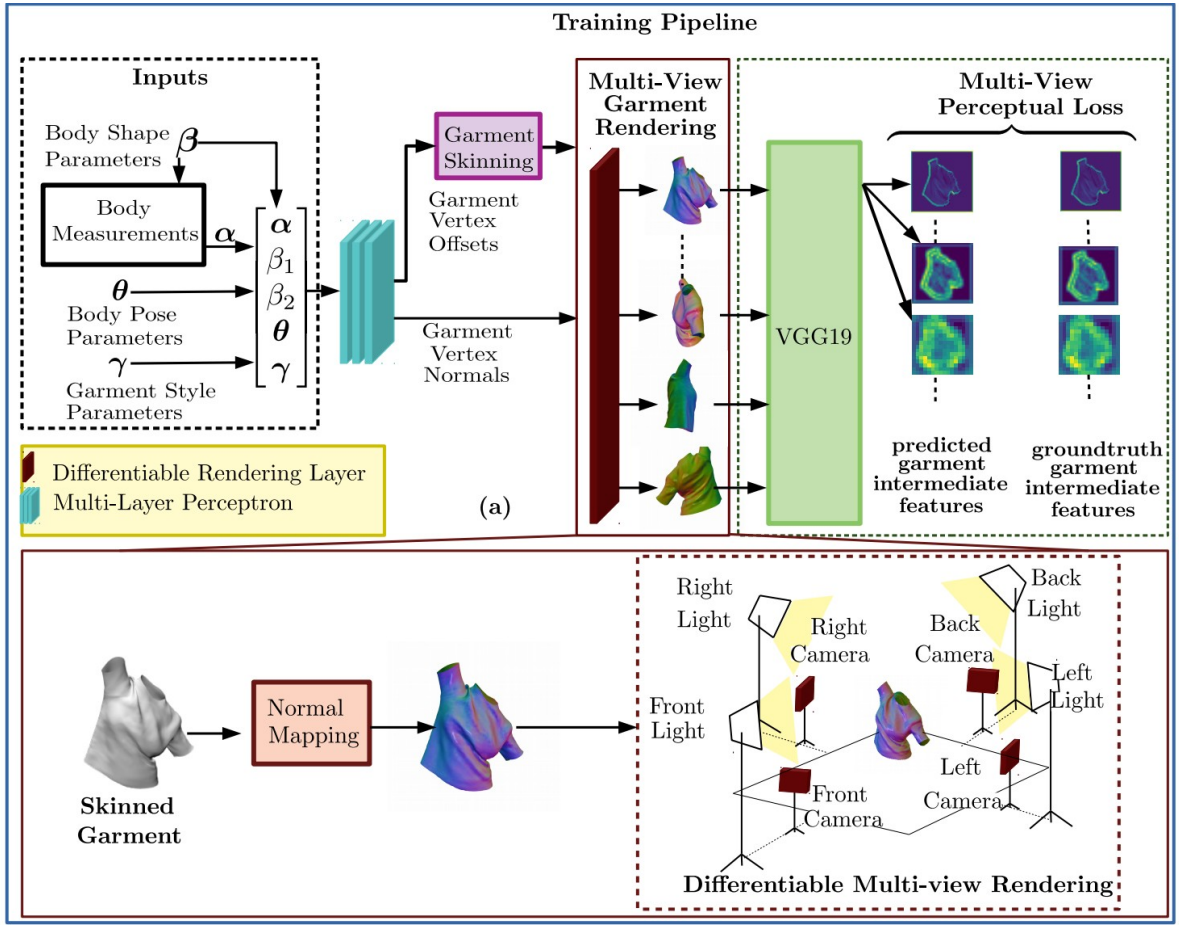
(b)

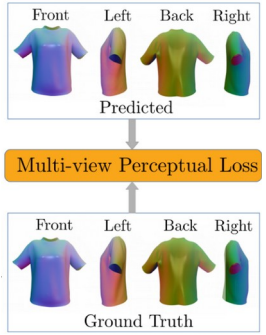


(c)

DeepDraper – Training Setup

Dataset : TailorNet dataset train-test split





DeepDraper – Training Losses

Geometric Losses

- L1-Loss on the predicted garment vertex offsets

$$\mathcal{L}_{OL} = \frac{1}{\kappa}\sum_{i=1}^{\kappa} ||\widehat{\mathbf{O}}_i - \mathbf{O}_i||_1$$

- Cosine similarity loss on the predicted garment vertex normals

$$\mathcal{L}_{NL} = \frac{1}{\kappa}\sum_{i=1}^{\kappa}(1 - \frac{\widehat{\mathbf{n}}_i \cdot \mathbf{n}_i}{||\widehat{\mathbf{n}}_i||||\mathbf{n}_i||})$$

- Regularization loss on the predicted garment vertex normals

$$\mathcal{L}_{NReg} = \frac{1}{\kappa}\sum_{i=1}^{\kappa} ||\widehat{\mathbf{n}}_i - \overline{\mathbf{n}}_i||_2$$

- Body-Garment collision loss

$$\mathcal{L}_{coll} = \frac{1}{\kappa}\sum_{i=1}^{\kappa} \max(-\mathbf{n}_j(\mathbf{v}_j - \widehat{\mathbf{g}}_i), \delta)$$

Perceptual Losses

- Perceptual metric as a loss on multi-view renderings. Weighted L1 loss between intermediate feature maps of VGG19

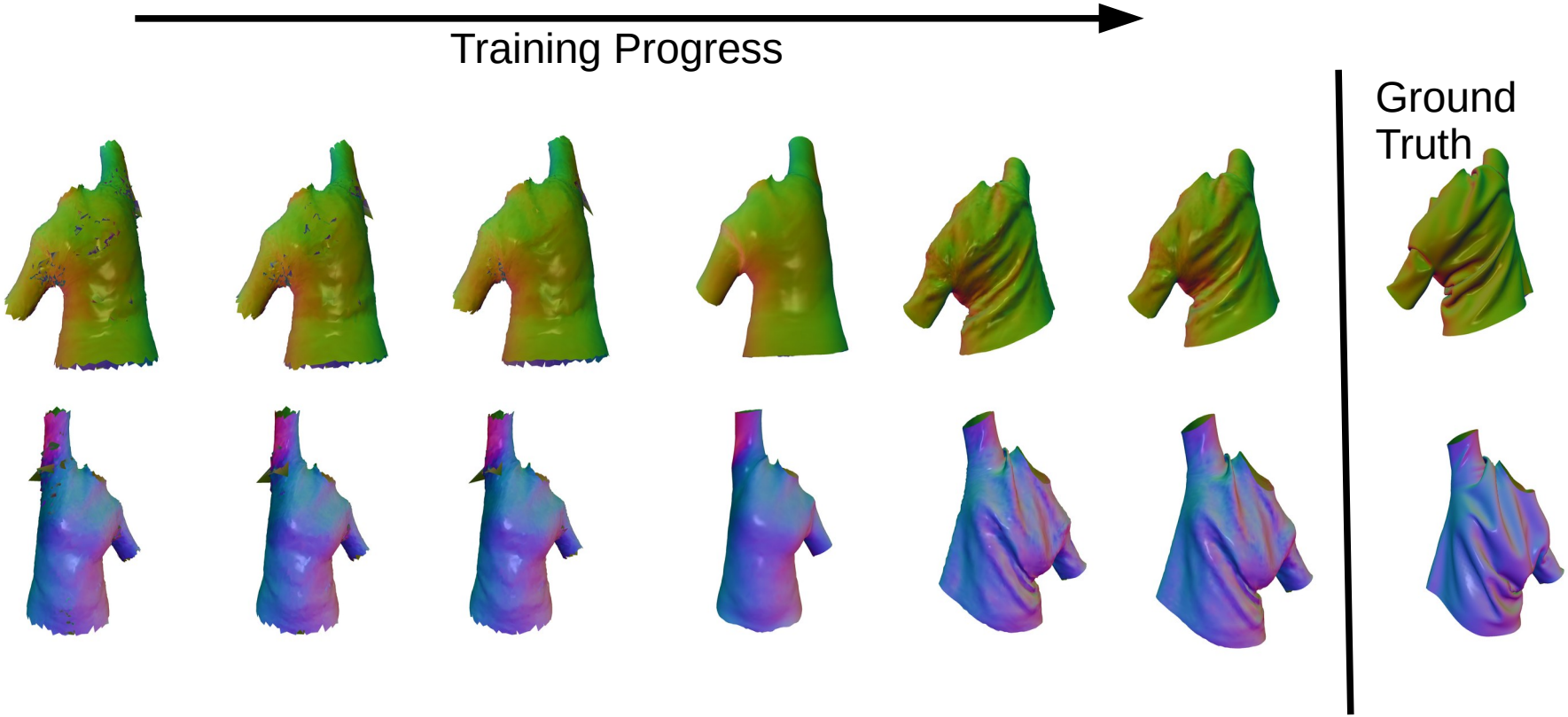
$$PL(I_{\widehat{\mathbf{G}},\mathbf{T},f}, I_{\widehat{\mathbf{G}},\widehat{\mathbf{T}},f}) = \sum_{l \in S} \lambda_l ||\Gamma^l_{\widehat{\mathbf{G}},\mathbf{T},f} - \Gamma^l_{\widehat{\mathbf{G}},\widehat{\mathbf{T}},f}||_1$$

- Content loss : L1 loss between predicted and GT rendered images

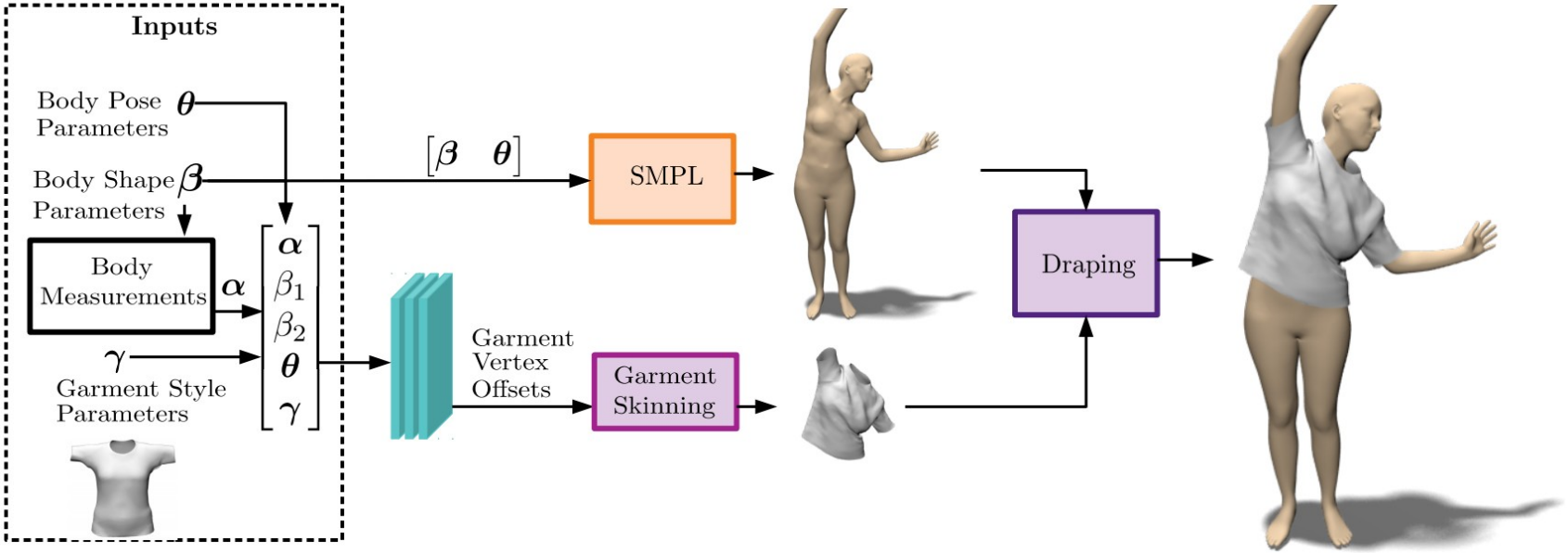
$$\ell^f_{con} = ||I_{\widehat{\mathbf{G}},\mathbf{T},f} - I_{\widehat{\mathbf{G}},\widehat{\mathbf{T}},f}||_1 + ||I_{\mathbf{G},\mathbf{T},f} - I_{\mathbf{G},\widehat{\mathbf{T}},f}||_1$$

[1] Zhang et.al, “The unreasonable effectiveness of deep features as a perceptual metric”, CVPR, 2018

DeepDraper – Training Progress (Intermediate Renderings)



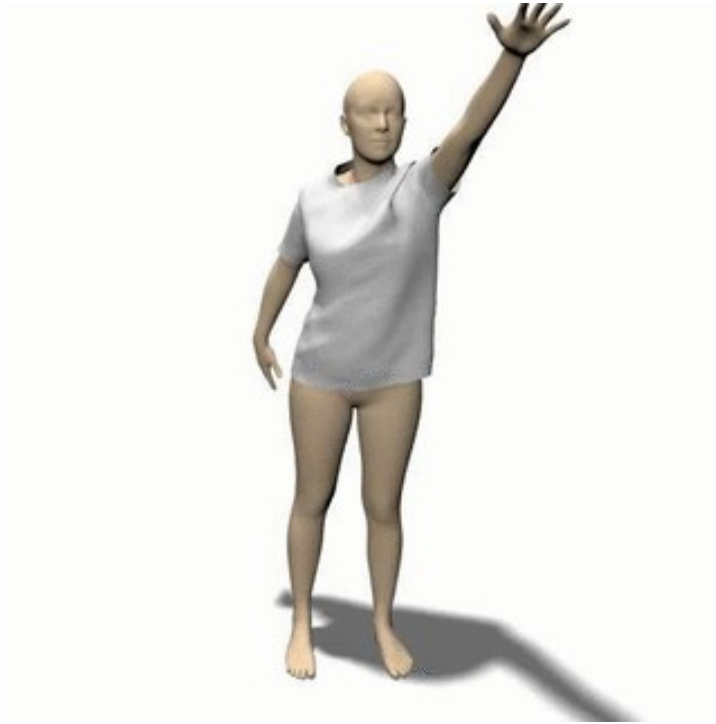
DeepDraper – Inference Setup



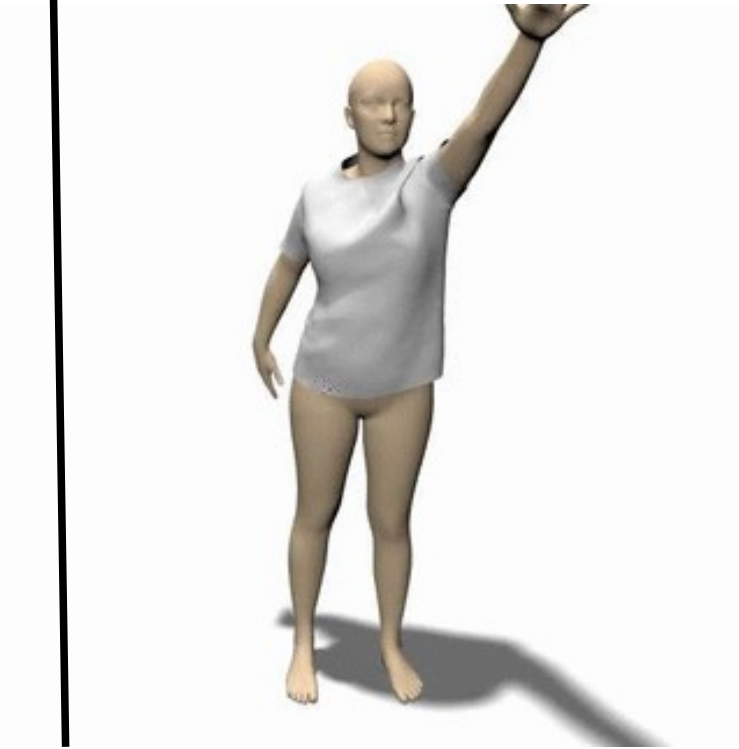
DeepDraper Results

Fixed T-shirt

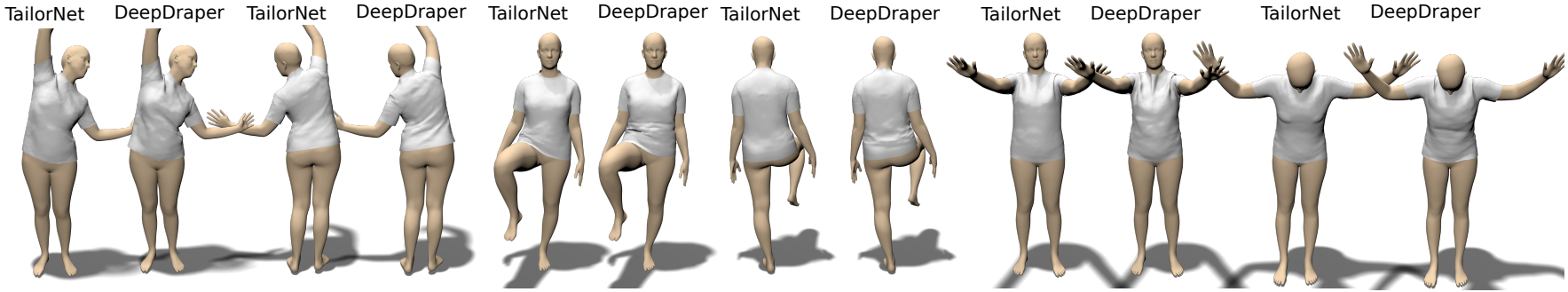
Varying overall body fatness

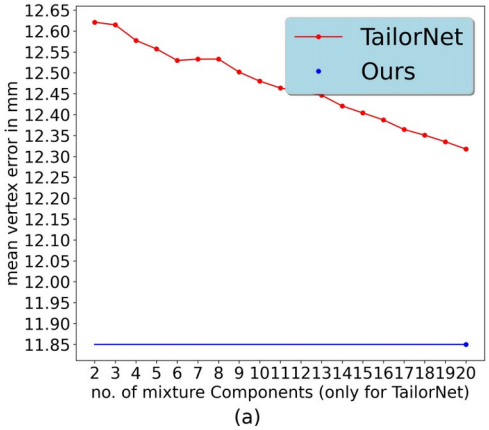


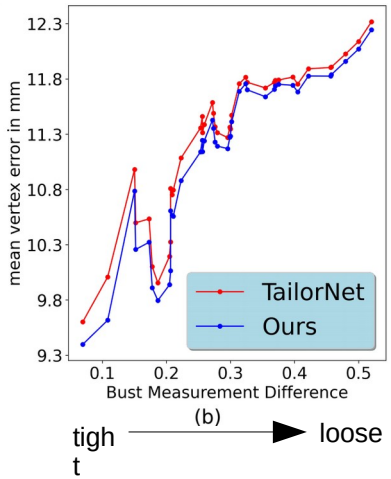
Varying overall body height

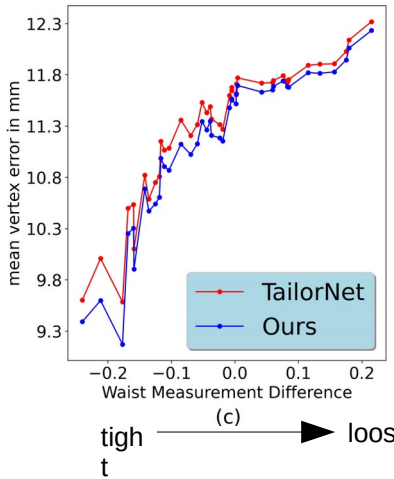


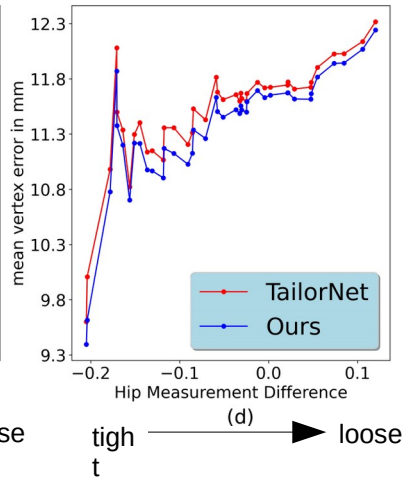
Analysis – Comparison with the TailorNet (CVPR 2020)









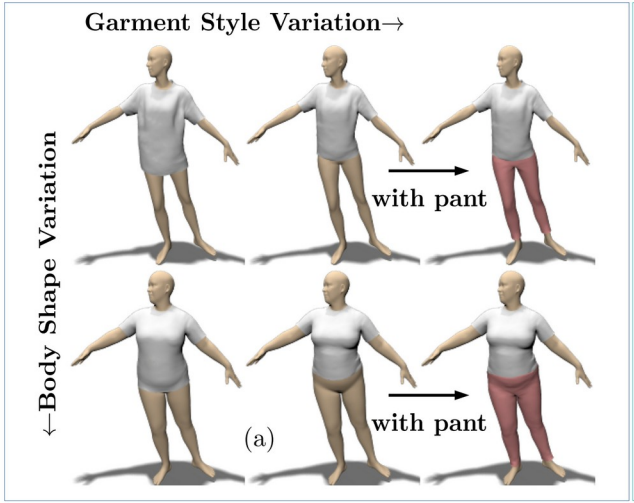


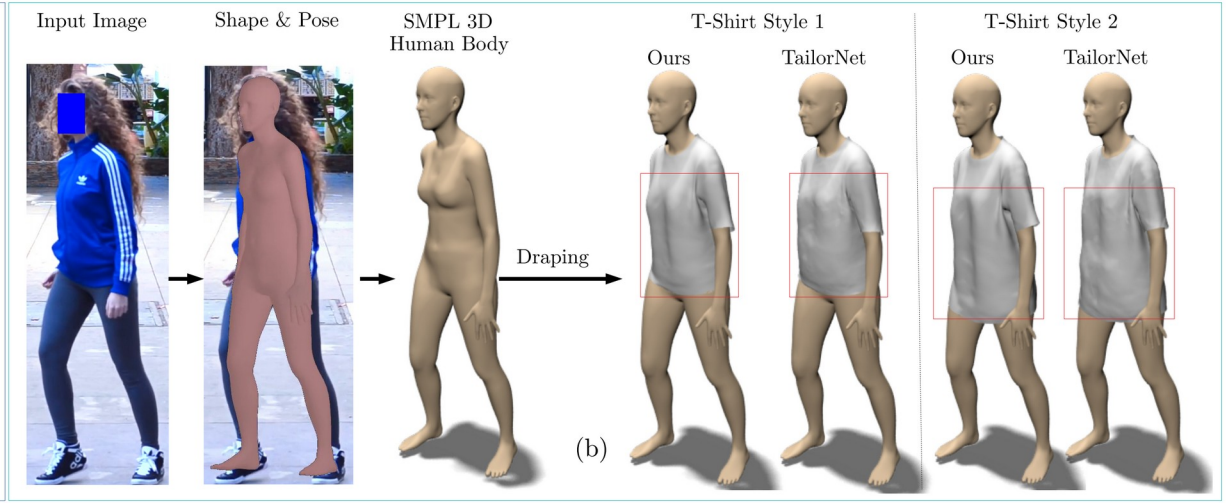
- TailorNet is sensitive to no. of mixture components
- Fails to generalize

DeepDraper is
23x faster, 10x smaller

Conclusion

- Training with coupled geometric and perceptual loss helps
- Explicitly inclusion of body measurements helps in predicting accurate fitting garments
- DeepDraper is fast and small in size, suitable for low end devices (mobile, tablets etc)
- Generalize well in the wild setting (unseen body shape, pose and garment styles)





Thank you