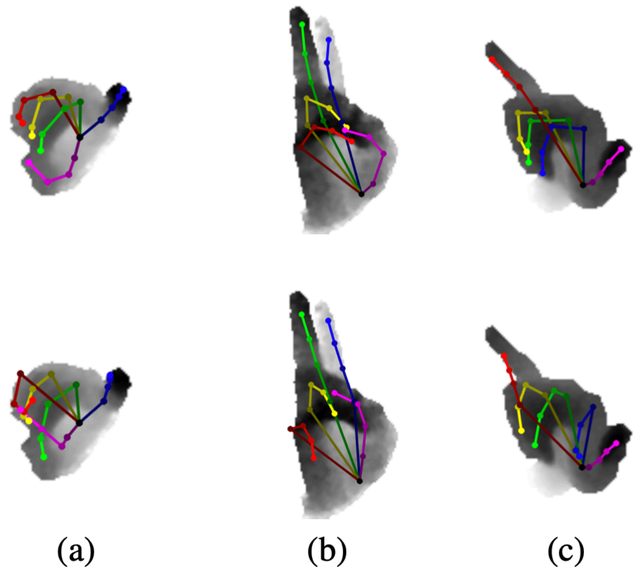
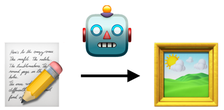
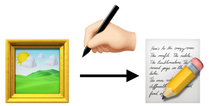
Hand-pose dataset annotation

Основные сложности

- Разметка point cloud очень дорогая
- Открытые датасеты представляют ограниченное количество возможных поз
- Существующие датасеты размечены не точно



Собираем свой датасет!

- Два возможных пути:
  - Генерация 
  - Сбор и разметка 

Efficiently Creating 3D Training Data for Fine Hand Pose Estimation

Обзор

Дан последовательность карт глубины с рукой в движении $\{\mathcal{D}_i\}_{i=1}^{N}$

Алгоритм:

- Найдем reference frames
- Человек размечает reference frames
- Распространение разметки на остальные карты
- Глобальная оптимизация

Efficiently Creating 3D Training Data for Fine Hand Pose Estimation

Reference frames

- Необходимо выбрать кадры которые будут «похожи» на те, которые находятся рядом в последовательности.
- $\max_{\mathcal{R}} f(\mathcal{R}) \quad \text{s.t.} \quad |\mathcal{R}| < M$
- R - набор выбранных reference frames
- f(R) - количество кадров, находящихся на выбранном расстоянии ρ хотя бы до одного из кадров в R.
- Хорошо решается жадным алгоритмом $f(e \cup \mathcal{R}) - f(\mathcal{R})$

Efficiently Creating 3D Training Data for Fine Hand Pose Estimation

3D Joint Locations in the Ref- erence Frames

- Будем делать 2D разметку
- $$\underset{\{L_{r,k}\}_{k=1}^{K}}{\arg\min} \sum_{k=1}^{K} v_{r,k} \| \text{proj}(L_{r,k}) - l_{r,k} \|_2^2 \qquad (4)$$

  $$\text{s.t.} \quad \forall k \ \ \| L_{r,k} - L_{r,p(k)} \|_2^2 = d_{k,p(k)}^2$$ — длины костей скелета соблюдены

  $$\forall k \ \ v_{r,k} = 1 \Rightarrow \mathcal{D}_r[l_{r,k}] < z(L_{r,k}) < \mathcal{D}_r[l_{r,k}] + \epsilon$$

  $$\forall k \ \ v_{r,k} = 1 \Rightarrow (L_{r,k} - L_{r,p(k)})^\top \cdot c_{r,k} > 0$$

  $$\forall k \ \ v_{r,k} = 0 \Rightarrow z(L_{r,k}) > \mathcal{D}_r[l_{r,k}]$$

- $L_{r,k}$ - трехмерное положение k-го сустава для r-го кадра
- $l_{r,k}$ - двумерная проекция, предоставленная человеком-аннотатором
- p(r) - возвращает индекс родительского сустава k-го сустава в скелете руки.
- Dr[lr,k] - значение глубины в Dr в точке lr,k. z(L) - глубина трехмерной точки L.

Efficiently Creating 3D Training Data for Fine Hand Pose Estimation

3D Joint Locations in the Remaining Frames

- Итеративно находим ближайший среди незамеченных и размечаем

$$\begin{bmatrix} \hat{c} \\ \hat{a} \end{bmatrix} = \underset{\substack{c \in [1;N] \setminus \mathcal{I} \\ a \in \mathcal{I}}}{\arg\min}\ d(\mathcal{D}_c, \mathcal{D}_a)\ .$$

$$\underset{\{L_{\hat{c},k}\}_k}{\arg\min} \sum_k \mathrm{ds}(\mathcal{D}_{\hat{c}}, \mathrm{proj}(L_{\hat{c},k}); \mathcal{D}_{\hat{a}}, l_{\hat{a},k})^2$$
$$\text{s.t.} \quad \forall k \quad \|L_{\hat{c},k} - L_{\hat{c},p(k)}\|_2^2 = d_{k,p(k)}^2\ ,$$

- $\mathrm{ds}(\mathcal{D}_1, \mathrm{proj}(L_1); \mathcal{D}_2, l_2)$ - различия между участком в D1 с центром на проекции proj(L1) и участком в D2 с центром на l2.
- Эта оптимизация ищет суставы, основываясь на их появлении в кадре $\hat{a}$, при этом соблюдаются трехмерные расстояния между суставами.

Efficiently Creating 3D Training Data for Fine Hand Pose Estimation

Global optimization

- C - сумма различий между суставами на не референсных кадрах
- TC - это простая модель движения 0-го порядка, которая обеспечивает временную гладкость 3D-локаций
- P - обеспечивает согласованность с человеческими 2D-аннотациями для референсных кадров

$$\sum_{i\in[1;N]\setminus\mathcal{R}}\sum_{k}\mathrm{ds}(\mathcal{D}_i,\mathrm{proj}(L_{i,k});\mathcal{D}_{\hat{i}},l_{\hat{i},k})^2 + \quad \text{(C)}$$

$$\lambda_M\sum_{i}\sum_{k}\|L_{i,k}-L_{i+1,k}\|_2^2 + \quad \text{(TC)}$$

$$\lambda_P\sum_{r\in\mathcal{R}}\sum_{k}v_{r,k}\|\mathrm{proj}(L_{r,k})-l_{r,k}\|_2^2 \quad \text{(P)}$$

$$\text{s.t.}\quad \forall i,k\quad \|L_{i,k}-L_{i,p(k)}\|_2^2=d_{k,p(k)}^2\ .$$

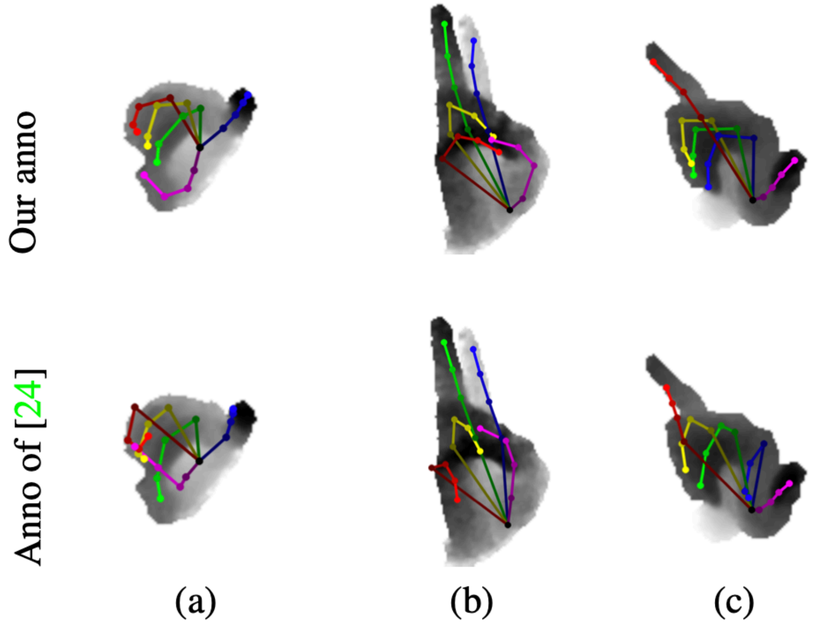
Efficiently Creating 3D Training Data for Fine Hand Pose Estimation

Результаты

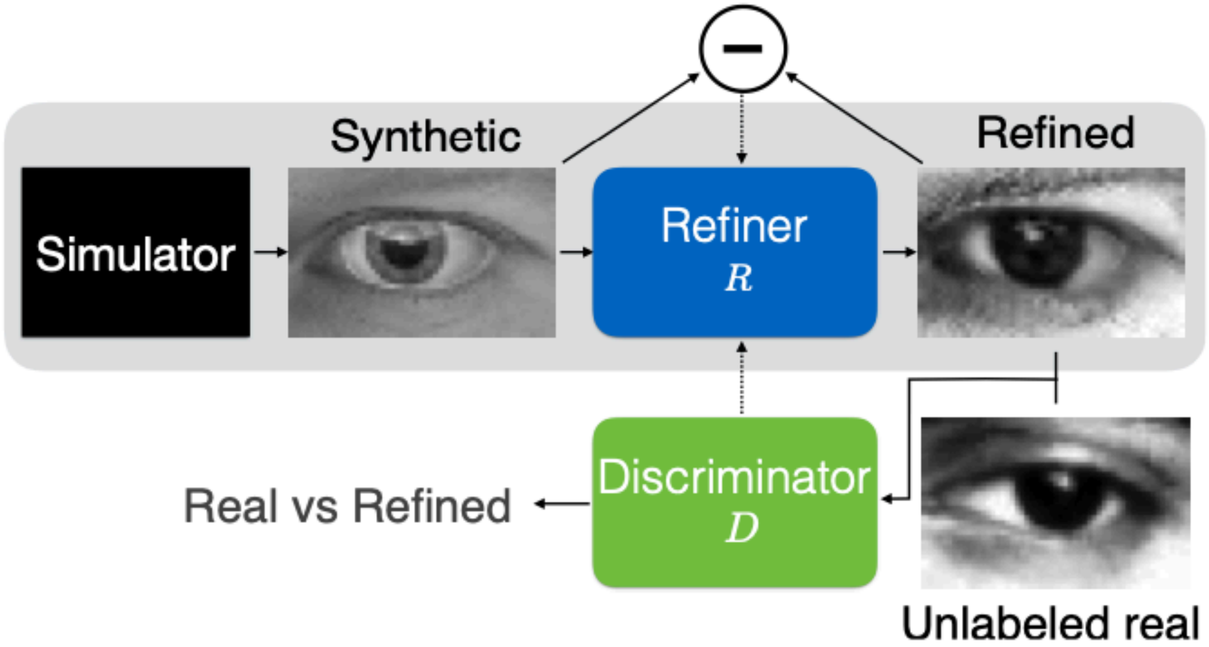
(a) Closest reference frame

(b) Initialization with SIFTFlow

(c) After optimization



Learning from Simulated and Unsupervised Images through Adversarial Training



Learning from Simulated and Unsupervised Images through Adversarial Training

Adversarial Loss with Self-Regularization

$$\mathcal{L}_R(\boldsymbol{\theta}) = \sum_i \ell_{\text{real}}(\boldsymbol{\theta}; \mathbf{x}_i, \mathcal{Y}) + \lambda \ell_{\text{reg}}(\boldsymbol{\theta}; \mathbf{x}_i),$$

$$\mathcal{L}_D(\boldsymbol{\phi}) = -\sum_i \log(D_{\boldsymbol{\phi}}(\tilde{\mathbf{x}}_i)) - \sum_j \log(1 - D_{\boldsymbol{\phi}}(\mathbf{y}_j)).$$

$$\ell_{\text{real}}(\boldsymbol{\theta}; \mathbf{x}_i, \mathcal{Y}) = -\log(1 - D_{\boldsymbol{\phi}}(R_{\boldsymbol{\theta}}(\mathbf{x}_i))).$$

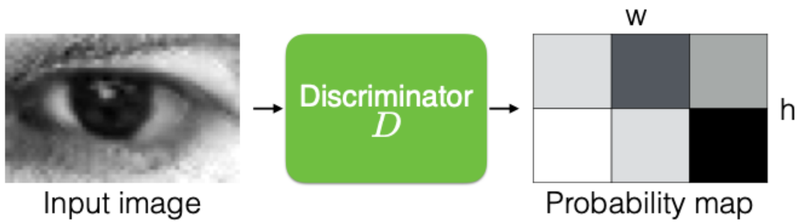
$$\ell_{\text{reg}} = \|\psi(\tilde{\mathbf{x}}) - \mathbf{x}\|_1$$

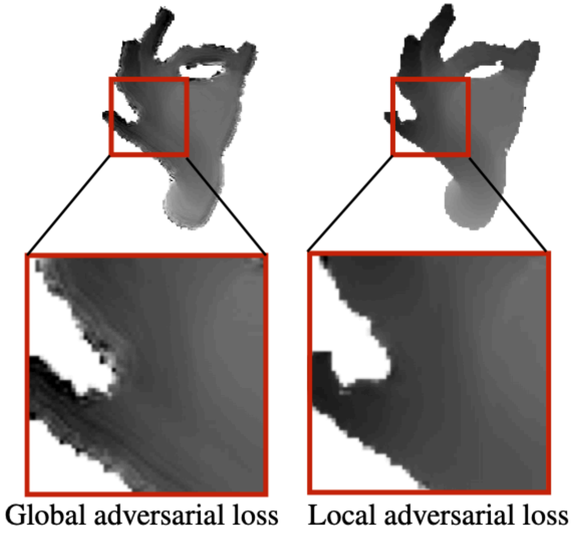
- Регуляризация минимизирует расстояние между признаками улучшенного и синтетического изображения
- Ψ - генерирует признаки изображений

Learning from Simulated and Unsupervised Images through Adversarial Training

Local Adversarial Loss

- Когда мы обучаем одну сильную сеть дискриминатора, refiner чрезмерно выделять определенные признаки изображения, что приводит к появлению артефактов
- Любой локальный участок, отобранный из refined изображения, должен иметь распределение, схожее с реальным участком изображения.
- Определим дескриминатор, который классифицирует все локальные участки изображения по отдельности

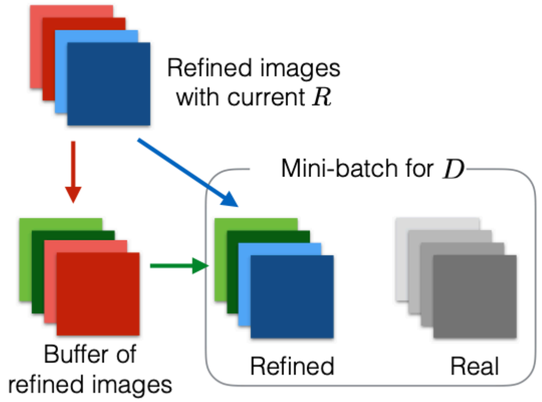


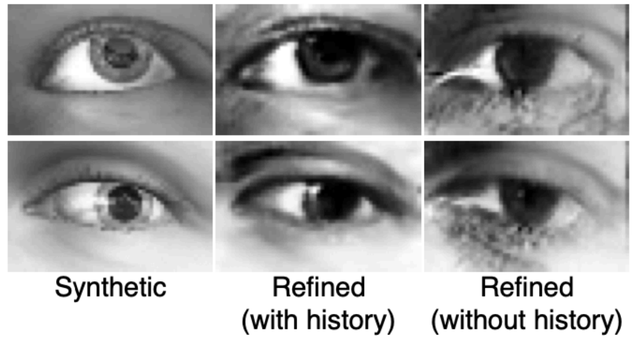


Learning from Simulated and Unsupervised Images through Adversarial Training

Updating Discriminator using a History of Refined Images

- Дискриминатор фокусируется только на последних refined изображениях
- Недостаток памяти может привести (i) к дивергенции в обучении генеративной сети и (ii) к тому, что сеть refiner снова выдаст артефакты, о которых забыл дискриминатор.





Learning from Simulated and Unsupervised Images through Adversarial Training

Результаты