A. Proof of Theorems and Propositions

A.1. Proof of Proposition 1

To show that $\mathcal{F}$ is path-invariant, it suffices to prove that $f_p = f_q$ for every path pair $(p, q) \in \mathcal{G}_{\text{pair}}$. But by Definition 7, (p, q) is either in $\mathcal{G}_{\text{pair}}$ or can be induced from a finite number of operations with merge, stitch and/or cut. So if we can show that the output path pair in every round of operation keeps consistency on the map network $\mathcal{F}$, given the input path pairs are consistent, then all path pairs on $\mathcal{G}$ would be path-invariant by employing an induction proof. Next we achieve this goal by considering three operations respectively.

- **merge.** The merge operation takes as input two path pairs $(p, q), (p', q') \in \mathcal{G}_{\text{pair}}$ where $p' = r \sim p \sim r'$, i.e., p' is formed by stitching three sub-paths r, p and r' in order. By Definition 2, it is easy to see that

  $$f_{p'} = f_{r'} \circ f_p \circ f_r.$$

  But we are given that $\mathcal{F}$ is consistent on the input pairs, or equivalently,

  $$f_p = f_q, \quad f_{p'} = f_{q'}.$$

  Hence

  $$f_{q'} = f_{p'} = f_{r'} \circ f_q \circ f_r = f_{r \sim q \sim r'}.$$

  So $\mathcal{F}$ is also consistent on path pair $(r \sim q \sim r', q')$.

- **stitch.** The stitch operation takes as input two path pairs $(p, q), (p', q')$ where $p, q \in \mathcal{G}_{\text{path}}(u, v)$ and $p', q' \in \mathcal{G}_{\text{path}}(v, w)$. Since $\mathcal{F}$ is consistent on (p, q) and (p', q'), it follows immediately

  $$f_{p \sim p'} = f_{p'} \circ f_p = f_{q'} \circ f_q = f_{q \sim q'},$$

  which means $\mathcal{F}$ is also consistent on $(p \sim p', q \sim q')$.

- **cut.** The cut operation takes as input two path pairs $(\mathcal{C}_1, \varnothing)$ and $(\mathcal{C}_2, \varnothing)$, where $\mathcal{C}_1$ and $\mathcal{C}_2$ are two common vertices u, v and share a common intermediate path from v to u. The two cycles can be represented by $u \xrightarrow{p} v \xrightarrow{q} u$ and $u \xrightarrow{p'} v \xrightarrow{q} u$ where $p, p' \in \mathcal{G}_{\text{path}}(u, v)$ and $q \in \mathcal{G}_{\text{path}}(v, u)$. Since $\mathcal{F}$ is consistent on $(\mathcal{C}_1, \varnothing)$ and $(\mathcal{C}_2, \varnothing)$, we have

  $$f_{p \sim q} = f_q \circ f_p = I, \quad f_{p' \sim q} = f_q \circ f_{p'} = I.$$

  However, it is known that the inverse of some function must be unique, giving the following result

  $$f_{p'} = f_p,$$

  or in other words, $\mathcal{F}$ is consistent on path pair (p, p').

The consistency of $\mathcal{F}$ on the output pairs for all three operations given the consistency on their input pairs ensures our proposition. □

A.2. Proof of Theorem 3.1

The algorithm adds exactly $|\mathcal{E}|$ edges in total. And during each edge insertion, at most $|\mathcal{P}| \leq |\mathcal{V}|$ path pairs would be added to $\mathcal{B}_{dag}(\sigma)$, thus it follows immediately that $|\mathcal{B}_{dag}(\sigma)| \leq |\mathcal{V}||\mathcal{E}|$.

Next we show that $\mathcal{B}_{dag}(\sigma)$ indeed is a path-invariance basis for $\mathcal{G}$. To this end, we will verify that every path pair in $\mathcal{G}_{\text{pair}}$ can be induced from a subset of $\mathcal{B}_{dag}(\sigma)$ by operations, using a induction proof. In particular, we claim that at all time points, all path pairs in $\mathcal{G}_{cur}$ can be induced from $\mathcal{B}_{dag}(\sigma)$ by a series of operations. Initially, this inductive assumption holds trivially since $\mathcal{G}_{cur}$ is an empty set.

Suppose now we were processing an edge $(u, v) \in \mathcal{E}$ (so $(u, v) \notin \mathcal{G}_{cur}(\sigma)$ at this time point) and let $\mathcal{G}'_{cur} = \mathcal{G}_{cur} \cup \{(u, v)\}$. By inductive assumption, all path pairs in $\mathcal{G}_{cur}$ can be induced from $\mathcal{B}_{dag}(\sigma)$. After inserting (u, v) into $\mathcal{G}_{cur}$, it suffices to consider path pairs that contain edge (u, v) since all other path pairs have been guaranteed by inductive assumption. Let (p, q) be a path pair in $\mathcal{G}'_{cur}$ containing (u, v). Without loss of generality, suppose $p, q \in \mathcal{G}_{\text{path}}(w, v)$ and

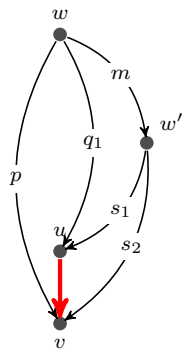


Figure 7: An Illustration for Path-Pair Generation

$$p = p_1 \sim (r, v), \; q = q_1 \sim (u, v).$$

If $r = u$, then (p, q) can be induced by stitching (p_1, q_1) and $((u, v), (u, v))$ where $(p_1, q_1) \in \mathcal{G}_{cur}$. We assume $r \neq u$, and then p would be a path from w to v in $\mathcal{G}_{cur}$ and q_1 would be a path from w to u in $\mathcal{G}_{cur}$.

Recall the definition of $\mathcal{P}$ and $\overline{\mathcal{P}}$. $w \in \mathcal{P}$ immediately follows. If $w \notin \overline{\mathcal{P}}$, then there exists $w' \neq w$ such that w can reach w' in $\mathcal{G}_{cur}$ and $w' \in \overline{\mathcal{P}}$ and denote such path as m. For convenience, we let $w' = w$ and $m = \varnothing$ when $w \in \overline{\mathcal{P}}$. Every vertex in $\overline{\mathcal{P}}$ corresponds to a path-invariance pair to be added to $\mathcal{B}_{dag}$ by our algorithm. Here we assume that it is $(s_1 \sim (u, v), s_2)$ for w' where $s_1 \in \mathcal{G}_{\text{path}}(w', u)$, $s_2 \in \mathcal{G}_{\text{path}}(w', v)$ and s_2 is within $\mathcal{G}_{cur}$.

By the property of DAG and the order of edge insertion, all paths from w to u in $\mathcal{G}$ are also in $\mathcal{G}_{cur}$ since $(u, v) \in \mathcal{E}$. Thus $(q_1, m \sim s_1)$ can be induced from $\mathcal{B}_{dag}(\sigma)$ by inductive assumption. Similarly, as s_2 is within $\mathcal{G}_{cur}$, $(p, m \sim s_2)$ is also a path-invariance pair, which can be induced from $\mathcal{B}_{dag}(\sigma)$. Next we give the operation steps

to build (p, q):

$$(m, m) + (s_1 \sim (u, v), s_2)$$
$$\xrightarrow{stitch} (m \sim s_1 \sim (u, v), m \sim s_2) \quad (6)$$
$$(m \sim s_1 \sim (u, v), m \sim s_2) + (q_1, m \sim s_1)$$
$$\xrightarrow{merge} (q_1 \sim (u, v), m \sim s_2) \quad (7)$$
$$(q_1 \sim (u, v), m \sim s_2) + (p, m \sim s_2)$$
$$\xrightarrow{merge} (p, q) \quad (8)$$

For the last step, notice that $q = q_1 \sim (u, v)$ and (p, q) is equivalent to (q, p). Thus all path pairs in $\mathcal{G}'_{cur}$ can be induced by path pairs in $\mathcal{B}_{dag}(\sigma)$ with a series of operations, which completes our proof by induction. □

A.3. Proof of Proposition 2

Before proving Proposition 2, we first introduce some well-known terms for depth-first search. There are two time stamps $d[v]$ and $f[v]$ for each vertex v, where d is defined as the time point when it visits v for the first time and f as the time point when it finishes visiting v. Some edge (u, v) in $\mathcal{E}$ can be classified into one of four disjoint types as follows:

- **Tree Edge:** v is visited for the first time as we traverse the edge (u, v). In this case (u, v) will be added into the resulting DFS spanning tree. For tree edge we have
$$d[u] < d[v], \quad f[u] > f[v].$$
- **Back Edge:** v is visited and is an ancestor of u in the current spanning tree. For back edge we have
$$d[u] > d[v], \quad f[u] < f[v].$$
- **Forward Edge:** u is visited and is an ancestor of v in the current spanning tree. For forward edge we have
$$d[u] < d[v], \quad f[u] > f[v].$$
- **Cross Edge:** v is visited and is neither an ancestor nor descendant of u in the current spanning tree. For cross edge we have
$$d[u] > d[v], \quad f[u] > f[v].$$

Using these definitions, we prove that:

Any cycle $\mathcal{C}$ in $\mathcal{G}$ have a vertex u in $\mathcal{C}$ such that all other vertices are located within the sub-tree rooted at u, i.e., are the descendants of u in $\mathcal{T}$.

Let
$$\mathcal{C} : u_1 \to u_2 \to \cdots \to u_k \to u_1.$$
Without loss of generality, u_1 is assumed to be the one with smallest d among all $\{u_i\}$. If not all u_i are descendants of u_1, we choose u_t to be the one with smallest t, which means u_{t-1} is a descendant of u_1 but u_t is not. Obviously (u_{t-1}, u_t) cannot be a tree edge or forward edge, which causes u_t to be a descendant of u_{t-1} and also a descendant of u_1. If (u_{t-1}, u_t) is a back edge, then u_t is not a descendant of u_1 if and only if $u_{t-1} = u_1$ since there is in fact unique back path in the spanning tree $\mathcal{T}$. But $u_{t-1} = u_1$ means u_t is a parent of u_1, and thus there exists a smaller d than u_1, which results in a contradiction. Also (u_{t-1}, u_t) cannot be a cross edge. In fact, since u_{t-1} is a descendant of u_1, we have $f[u_{t-1}] < f[u_1]$. Together with $f[u_t] < f[u_{t-1}]$ from cross edge property, we have $f[u_t] < f[u_1]$. But u_t is not a descendant or ancestor of u_1, which means the sub-tree rooted at u_1 is disjoint from the sub-tree rooted at u_t, so intervals $[d[u_1], f[u_1]]$ and $[d[u_t], f[u_t]]$ must be disjoint by the property of depth-first search. As thus $f[u_t] < f[u_1]$ implies $d[u_t] < d[u_1]$, which contradicts the assumption that $d[u_1]$ is smallest among u_i. Hence all u_i are descendants of u_1.

Now come back to the original proposition. Continue using the notation $\mathcal{C}$ defined above. In addition we define $\mathcal{C}_i$ as the sub-path from u_1 to u_i, i.e.,
$$\mathcal{C}_i : u_1 \to u_2 \to \cdots \to u_i.$$

We will show $\mathcal{C}$ can be induced from $\mathcal{B}$ by a finite number of operations with merge, stitch and cut. Above all, we have assumed the property of path-invariance on $\mathcal{T}$ by Theorem 3.1. Given u_1 is the common ancestor of all u_i, we inductively prove the following statement:

The path $u_1 \to u_2 \to \cdots \to u_t$ $(t \le k)$ is equivalent to the tree path from u_1 to u_t. Here tree path means a path in which all edges are in the spanning tree $\mathcal{T}$.

The base case is trivial. Now suppose $u_1 \to u_2 \to \cdots \to u_t$ $(t < k)$ is equivalent to tree path P from u_1 to u_t and we continue to check $u_1 \to u_2 \to \cdots \to u_{t+1}$.

- If (u_t, u_{t+1}) is a tree edge, then $P \sim (u_t, u_{t+1})$ is still a tree path and a stitch operation on path pair $(\mathcal{C}_t, P)$ and $((u_t, u_{t+1}), (u_t, u_{t+1}))$ gives the equivalency that we want.
- If (u_t, u_{t+1}) is a forward edge, then there exists a tree path P_1 from u_t to u_{t+1}. By path-invariance on $\mathcal{G}_{dag}$, we can stitch two path-invariance pair $(\mathcal{C}_t, P)$ and $((u_t, u_{t+1}), P_1)$ to obtain the desired equivalency.
- If (u_t, u_{t+1}) is a back edge, then there exists a tree path P_1 from u_{t+1} to u_t. In addition by our construction the cycle $P_1 \sim (u_t, u_{t+1})$ has been added into our basis set $\mathcal{B}$. Denote the tree path from u_1 to u_{t+1} as P_2, then stitching (P_2, P_2) and $(P_1 \sim (u_t, u_{t+1}), \varnothing)$ gives $(P_2 \sim P_1 \sim (u_t, u_{t+1}), P_2)$. On the other hand, by inductive assumption we have path-invariance pair $(P_2 \sim P_1, \mathcal{C}_t)$ since $P_2 \sim P_1$ is just the tree path from u_1 to u_t. Thus by merging $(P_2 \sim P_1, \mathcal{C}_t)$ and $(P_2 \sim P_1 \sim (u_t, u_{t+1}), P_2)$ we obtain the path pair $(\mathcal{C}_t \sim (u_t, u_{t+1}), P_2)$, or equivalently, $(\mathcal{C}_{t+1}, P_2)$.

- If (u_t, u_{t+1}) is a cross edge, then (u_t, u_{t+1}) has been included in $\mathcal{G}_{dag}$. Denote by P_1 the tree path from u_1 to u_t. In this way all $P_1 \sim (u_t, u_{t+1})$ would be equivalent to another tree path P_2 from u_1 to u_{t+1} since all edges involved here are within $\mathcal{G}_{dag}$ which maintains all possible path-invariance pairs. By merging path pairs $(P_1 \sim (u_t, u_{t+1}), P_2)$ and $(P_1, \mathcal{C}_t)$ we obtain path pair $(\mathcal{C}_t \sim (u_t, u_{t+1}), P_2)$, or $(\mathcal{C}_{t+1}, P_2)$, which is exactly we want to verify.

As thus we finished our inductive proof. In particular, the path (also a cycle) $u_1 \to \cdots \to u_k \to u_1$ is equivalent to $\varnothing$, or more precisely, the path pair $(\mathcal{C}, \varnothing)$ can be induced from $\mathcal{B}$ by a finite number of merge and stitch operations.

To complete our proof, we need to show that all path pairs in $\mathcal{G}$ instead of just $\mathcal{G}_{dag}$ can be induced from $\mathcal{B}$. This is relatively easy. Consider two path P_1 and P_2 both from u to v. Since $\mathcal{G}$ is strongly connected, there must exist some path P_3 from v to u. The cut operation on $P_1 \sim P_3$ and $P_2 \sim P_3$ for the common vertices u and v immediately gives the path pair (P_1, P_2). □

A.4. Proof of Theorem 3.2

To prove this theorem, we first prove the following lemma:

Lemma *Suppose $\mathcal{G}_i$ and $\mathcal{G}_j$ are two strongly connected components in $\mathcal{G}$ with $(\mathcal{G}_i, \mathcal{G}_j) \in \mathcal{G}_{dag}$. Given any vertices $u, u' \in \mathcal{G}_i$ and $v, v' \in \mathcal{G}_j$ with $(u, v), (u', v') \in \mathcal{E}_{ij}$, and paths $p \in \mathcal{G}_{\text{path}}(u, u')$, $p' \in \mathcal{G}_{\text{path}}(v, v')$, we claim that $p \sim (u', v')$ is equivalent to $(u, v) \sim p'$ under $\mathcal{B}$.*

In fact since $\mathcal{B}$ ensures equivalence for all path pairs inside the same SCC, the specific p, p' does not matter. We only care about the starting and ending points when everything happens inside a single SCC. So in the following proof we will use $P(x, y)$ to denote some path from x to y inside the single SCC but not mentioning the intermediate vertices. Recall that we built a (undirected) spanning tree $\mathcal{T}$ on $\mathcal{E}_{ij}^2$. Thus we have an edge sequence

$$(u_1, v_1), (u_2, v_2), \ldots, (u_k, v_k)$$

where $u_1 = u$, $v_1 = v$, $u_k = u'$, $v_k = v'$, and edge pair

$$((u_l, v_l), (u_{l+1}, v_{l+1}))$$

are in $\mathcal{T}$ for all $l = 1, \ldots, k-1$. Next we inductively prove that $P(u_1, u_t) \sim (u_t, v_t)$ is equivalent to $(u_1, v_1) \sim P(v_1, v_t)$ for $t = 1, \ldots, k$. The base case where $t = 1$ is trivial. Given the correctness for t, consider $t+1$. It is known that $(u_t, v_t) \sim P(v_t, v_{t+1})$ is equivalent to $P(u_t, u_{t+1}) \sim (u_{t+1}, v_{t+1})$ by the construction of $\mathcal{T}$ and $P(u_1, u_t) \sim (u_t, v_t)$ is equivalent to $(u_1, v_1) \sim P(v_1, v_t)$ by inductive assumption. By successively applying two merge operations on path

$$(u_1, v_1) \sim P(v_1, v_t) \sim P(v_t, v_{t+1})$$

we obtain the equivalent path

$$P(u_1, u_t) \sim P(u_t, u_{t+1}) \sim (u_{t+1}, v_{t+1}).$$

But it is straightforward that $P(u_1, u_t) \sim P(u_t, u_{t+1})$ is equivalent to $P(u_1, u_{t+1})$ under $\mathcal{B}$ since u_1, u_t, u_{t+1} are in the same SCC. Similarly, $P(v_1, v_t) \sim P(v_t, v_{t+1})$ is equivalent to $P(v_1, v_{t+1})$. Thus finally we obtain the equivalency on $P(u_1, u_{t+1}) \sim (u_{t+1}, v_{t+1})$ and $(u_1, v_1) \sim P(v_1, v_{t+1})$, which completes our inductive proof and the lemma immediately follows.

Come back to the original theorem. With notation $P(x, y)$, we can express an arbitrary path p in $\mathcal{G}$ from u to v as

$$p : P(u_1, v_1) \sim (v_1, u_2) \sim P(u_2, v_2) \sim \cdots \sim (v_{k-1}, u_k) \sim P(u_k, v_k)$$

where $u_1 = u$, $v_k = v$, and u_i, v_i are in the same SCC $\mathcal{G}_{b_i}$. Similarly write another path p' from u to v this way:

$$p' : P(u'_1, v'_1) \sim (v'_1, u'_2) \sim P(u'_2, v'_2) \sim \cdots \sim (v'_{k-1}, u'_k) \sim P(u'_k, v'_k)$$

where $u'_1 = u$, $v'_k = v$, and u_i, v_i are in the same SCC $\mathcal{G}_{b'_i}$ with obvious constraints $b_1 = b'_1$ and $b_k = b'_k$. As we extend $\mathcal{B}_{dag}$ that maintains the equivalency on all possible pairs in $\mathcal{G}_{dag}$ to $\mathcal{G}$, there would be a path pair

$$q : P(\alpha_1, \beta_1) \sim (\beta_1, \alpha_2) \sim P(\alpha_2, \beta_2) \sim \cdots \sim (\beta_{k-1}, \alpha_k) \sim P(\alpha_k, \beta_k)$$
$$q' : P(\alpha'_1, \beta'_1) \sim (\beta'_1, \alpha'_2) \sim P(\alpha'_2, \beta'_2) \sim \cdots \sim (\beta'_{k-1}, \alpha'_k) \sim P(\alpha'_k, \beta'_k)$$

in the extended $\mathcal{B}_{dag}$ where $\alpha_1 = \alpha'_1$, $\beta_k = \beta'_k$, and α_i, β_i are in the same SCC $\mathcal{G}_{b_i}$ while α'_i, β'_i in $\mathcal{G}_{b'_i}$. Thus it suffices to prove that p is equivalent to $P(u_1, \alpha_1) \sim q \sim P(\beta_k, v_k)$ while p' equivalent to $P(u'_1, \alpha'_1) \sim q' \sim P(\beta'_k, v'_k)$. (Recall that $u'_1 = u_1$, etc.) Since the proofs for them are essentially identical, we only consider p.

In fact, $P(u_1, \alpha_1) \sim q \sim P(\beta_k, v_k)$ can be equivalently expressed as

$$\begin{aligned} P(u_1, v_1) &\sim P(v_1, \beta_1) \sim (\beta_1, \alpha_2) \sim P(\alpha_2, u_2) \\ &\sim P(u_2, v_2) \sim P(v_2, \beta_2) \sim \ldots \\ &\sim P(\beta_{k-1}, \alpha_k) \sim (\alpha_k, u_k) \sim P(u_k, v_k). \end{aligned}$$

In other words, we split $P(\alpha_i, \beta_i)$ into $P(\alpha_i, u_i) \sim P(u_i, v_i) \sim P(v_i, \beta_i)$ for $i = 2, \ldots, k-1$. However, our lemma just states that

$$P(v_i, \beta_i) \sim (\beta_i, \alpha_{i+1})$$

is equivalent to

$$(v_i, u_{i+1}) \sim P(u_{i+1}, \alpha_{i+1}).$$

Figure 8: Visual comparison between our approach and state-of-the-art approaches. This figure is best viewed in color, zoomed in.

Thus by series of merge operations, $P(u_1, \alpha_1) \sim q \sim P(\beta_k, v_k)$ can be shown to be equivalent to

$$\begin{aligned} &P(u_1, v_1) \sim (v_1, u_2) \sim P(u_2, \alpha_2) \sim P(\alpha_2, u_2) \\ &\sim P(u_2, v_2) \sim P(v_2, u_3) \sim \dots \\ &\sim P(u_k, \alpha_k) \sim P(\alpha_k, u_k) \sim P(u_k, v_k), \end{aligned}$$

which is clearly p by cancelling all consecutive P's. □

A.5. Proof of Proposition 3

First note that in fact the bound $|\mathcal{V}||\mathcal{E}|$ theorem 3.1 can be improved to $(|\mathcal{V}| - 1)|\mathcal{E}|$ since $|\mathcal{P}| \leq |\mathcal{V}| - 1$ all time.

In this way the size of $\mathcal{B}_i$ is bounded by $|\mathcal{E}(\mathcal{B}_i)|(|\mathcal{V}(\mathcal{B}_i)| - 1)$. Suppose there are k strongly connected components in $\mathcal{G}$ and c edges across different SCCs. Then there are at most c edges in $\bigcup_{i,j} \mathcal{B}_{ij}$ since the edge number of a spanning tree is less than that of vertices by 1. Notice c is also the edge number of contracted graph $\mathcal{G}_{dag}$. Hence for the $\mathcal{G}_{dag}$, there are would be at most $(k-1) \times c$ items in $\mathcal{B}_{dag}$. Also observe that each SCC can have at most $|\mathcal{V}| - k + 1$ vertices when there are k SCCs. So the size of $\mathcal{B}$ would be bounded by

$$\begin{aligned} &(k-1)c + c + (|\mathcal{V}| - k)|\mathcal{E}| \\ \leq &k|\mathcal{E}| + (|\mathcal{V}| - k)|\mathcal{E}| \\ = &|\mathcal{V}||\mathcal{E}| \end{aligned}$$

□

B. Additional Details of Joint Dense Image Map

B.1. Training Details

We applied ADAM [27] to solve the following optimization problem for predicting dense image correspondences.

$$\min_{\theta} \sum_{(i,j)\in\mathcal{E}} \|f_{ij}^{\theta} - f_{ij}^{in}\|_1 + \lambda \sum_{(p,q)\in\mathcal{B}} \|f_p^{\theta} - f_q^{\theta}\|_{\mathcal{F}}^2 \quad (9)$$

We initialize f^{θ} by directly fitting it to the input image flows between pairs of images. We then impose the path-invariance regularization term to improve the network flow.

B.2. More Qualitative Evaluations

Figure 8 and Figure 9 provide more qualitative evaluations of our approach on the PASCAL Rigid categories. Besides the two categories shown in the main paper (Car and Aeroplane), we pick one example from the remaining 10 rigid categories. Note that our approach is consistently better than all baseline approaches.

B.3. Comparison to [56]

We run additional experiments to compare our approach against [56] (using DSP as input) on the Car dataset, where we used 1000 additional synthetic images. Table 2 shows the performance of different baselines. Our approach is superior to [56]. The improvement comes from using the network to propagate learned flows between similar images. Note that [56] essentially enforces consistency among selected 4-cycles (two synthetic and two real), so its performance is similar to Ours-Dense, which involves 3-cycles.

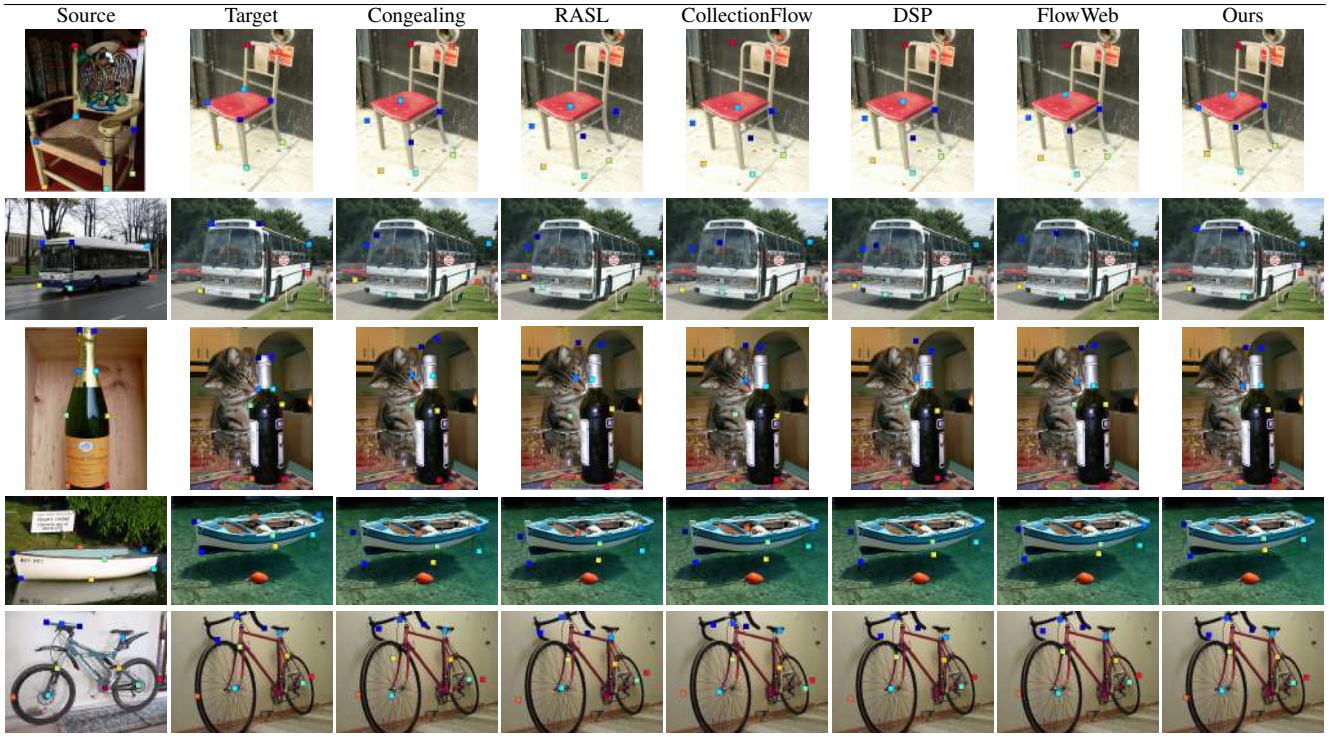


Figure 9: Visual comparison between our approach and state-of-the-art approaches. This figure is best viewed in color, zoomed in.

	Ours (Real-only)	[56]	Ours-Dense	Ours
PCK	0.65	0.59	0.60	0.68

Table 2: Additional comparison to [56].

C. Additional Details of 3D Semantic Scene Segmentation

C.1. Network Architecture and Training Details

For point cloud semantic segmentation network, we follow the same configuration from PointNet++ [37]. For voxel semantic segmentation network, we use the same network architecture proposed in 3D U-Net[9]. To generate training data from ScanNet scenes, following PointNet++ [37], we sample 1.5m by 1.5m by 3m cubes from the initial scenes. We sample such training cubes on the fly and randomly rotate each sample along the up-right axis. During test time, we split the test scene into smaller cubes first, and then merge label prediction in all the cubes from the same scene. Note that this is done for the prediction using each 3D representation in isolation.

We applied ADAM [27] to solve the optimization problem for predicting semantic labels in 3D scenes. We first initialize network parameters using the pre-trained weight on labeled data, and then impose the path-invariance regularization term to improve the network performance.

C.2. More Quantitative Evaluations

Table 3 shows per-class semantic voxel label prediction accuracy on ScanNet [12] test scenes. Compared to baseline methods, our approach shows consistently better performance compared to using 8% labeled data, and competitive results compared to using 30% and 100% labeled data, especially on frequently appeared classes, such as floor, wall, chair, sofa, and etc.

C.3. More Qualitative Evaluations

Figure 10 presents more qualitative comparisons between our approach and baselines. Consistently, using 8% labeled data and 92% unlabeled data, our approach achieved competing performance as using 30% to 100% labeled data when trained on each individual representation, and better performance as using 8% labeled data.

D. Additional Details of Joint Shape Matching

D.1. Training Details

We applied ADAM [27] to solve the following optimization problem:

$$\sum_{(i,j)\in\mathcal{E}} \|X_{ij} - X_{ij}^{in}\|_1 + \lambda \sum_{(p,q)\in\mathcal{B}} \|X_p - X_q\|_{\mathcal{F}}^2 \quad (10)$$

Initially, we set $X_{ij} = X_{ij}^{in}$. We also tried reweighted nonlinear least squares and used Gauss-Newton optimization

	Floor	Wall	Chair	Sofa	Table	Door	Cabinet	Bed	Desk	Toilet	Sink	Window	Picture	BookSh	Curtain	ShowerC	Counter	Fridge	Bathtub	OtherF	Total
Weight	35.7	38.8	3.8	2.5	3.3	2.2	2.4	2.0	1.7	0.2	0.2	0.4	0.2	1.6	0.7	0.04	0.6	0.3	0.2	2.9	-
PCI	90.9	98.1	58.4	45.4	40.2	47.4	36.4	62.8	21.8	35.4	32.0	16.7	21.5	0.0	0.0	1.3	0.0	0.0	19.7	9.6	79.2
	93.3	98.4	70.3	54.8	50.0	49.2	80.9	87.1	18.4	83.7	58.9	8.4	0.2	1.0	1.8	2.9	3.7	0.0	13.0	5.7	82.3
	88.0	97.8	76.3	62.7	19.9	63.5	65.5	59.7	52.5	63.9	76.2	17.4	27.1	17.0	12.2	56.1	0.0	0.0	25.7	22.0	80.8
	90.8	98.2	78.0	67.5	42.8	74.8	79.6	79.8	58.2	78.0	82.1	53.1	42.3	12.1	28.2	70.0	52.7	0.0	37.3	18.7	84.2
PCII	91.5	97.2	49.4	32.2	32.4	44.3	30.8	70.1	24.9	45.0	35.0	29.2	23.9	0.0	10.6	1.1	0.0	0.0	18.0	10.0	78.3
	94.9	98.4	65.0	58.1	48.0	41.7	65.4	89.6	31.2	81.0	62.9	4.6	4.6	0.0	0.4	3.7	0.0	0.0	17.5	4.5	82.5
	90.8	98.5	74.4	54.6	34.4	49.3	46.7	77.3	39.3	74.8	71.9	22.8	35.6	0.0	0.0	24.8	0.0	0.0	25.4	11.7	81.9
	92.8	98.0	86.4	64.2	29.8	55.0	59.2	75.3	37.6	86.5	67.6	9.3	25.3	23.5	19.0	46.6	43.1	0.0	25.0	13.7	83.3
PCIII	92.7	96.7	73.3	52.9	16.7	36.4	1.3	55.7	12.1	27.0	27.1	16.6	11.5	0.0	0.2	8.9	0.0	0.0	15.0	1.6	78.4
	93.7	98.1	71.4	58.9	50.0	54.4	59.9	74.8	30.6	82.8	65.1	10.6	1.6	1.4	0.8	21.5	0.0	0.0	20.3	8.7	82.3
	90.8	98.5	74.4	54.6	34.4	49.3	46.7	77.3	39.3	74.8	71.9	22.8	35.6	0.0	0.0	24.8	0.0	0.0	25.4	11.7	81.2
	90.4	97.6	76.1	65.0	45.5	80.6	70.9	75.3	32.4	82.0	73.9	48.0	49.8	13.5	16.9	64.4	46.7	0.0	42.0	13.0	83.4
VOLI	93.4	97.3	71.9	68.0	16.2	0.2	0.0	58.1	34.3	25.1	3.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	18.3	8.6	78.7
	93.5	97.6	70.7	61.2	55.7	39.1	55.0	76.7	11.5	81.3	68.8	0.3	2.3	2.2	0.0	2.0	0.0	0.0	16.8	10.2	81.6
	94.0	97.6	68.0	68.2	16.7	41.2	0.0	75.1	0.0	70.2	30.4	0.0	0.0	0.4	0.0	0.0	0.0	0.0	24.9	6.9	80.3
	92.5	97.5	74.2	67.2	25.0	55.0	59.5	62.9	0.0	85.4	0.0	3.9	38.5	0.4	0.0	0.0	42.5	0.0	37.8	14.2	81.9
VOLII	94.8	97.5	56.0	0.0	42.3	19.8	28.3	57.3	9.7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	13.5	5.1	77.4
	92.8	97.7	69.6	0.0	53.8	31.6	66.4	68.2	11.4	77.3	0.0	0.0	0.0	0.0	1.1	0.0	0.0	0.0	19.8	9.8	79.0
	92.5	98.1	62.4	54.4	15.3	50.0	0.0	59.1	0.0	74.5	61.9	0.0	0.0	0.0	0.0	0.0	0.0	0.0	39.0	9.3	79.5
	91.0	96.9	68.4	60.5	31.4	59.1	70.0	81.2	0.0	86.3	0.0	11.1	0.0	0.0	1.4	0.0	0.0	0.0	50.1	15.1	81.5

Table 3: Per-class semantic voxel label prediction accuracy on ScanNet test scenes. All numbers are in percentages. The first row indicates the percentages of each class in all test scenes, and then for 4 rows in each representation, we show the per-class prediction accuracy in 4 configurations: 8% Label, 8% Label + 92%Unlabel, 30% Label and 100% Label. (BookSh, ShowerC and OtherF are short Bookshelf, Shower Curtain and Other furniture, respectively.)

to solve the induced non-linear least square problem (we used conjugate gradient to solve the induced linear system) . We found that the optimal solutions of both approaches are similar, suggesting both of them reached a strong local minimum. Computationally, we find the ADAM optimizer to be more efficient.

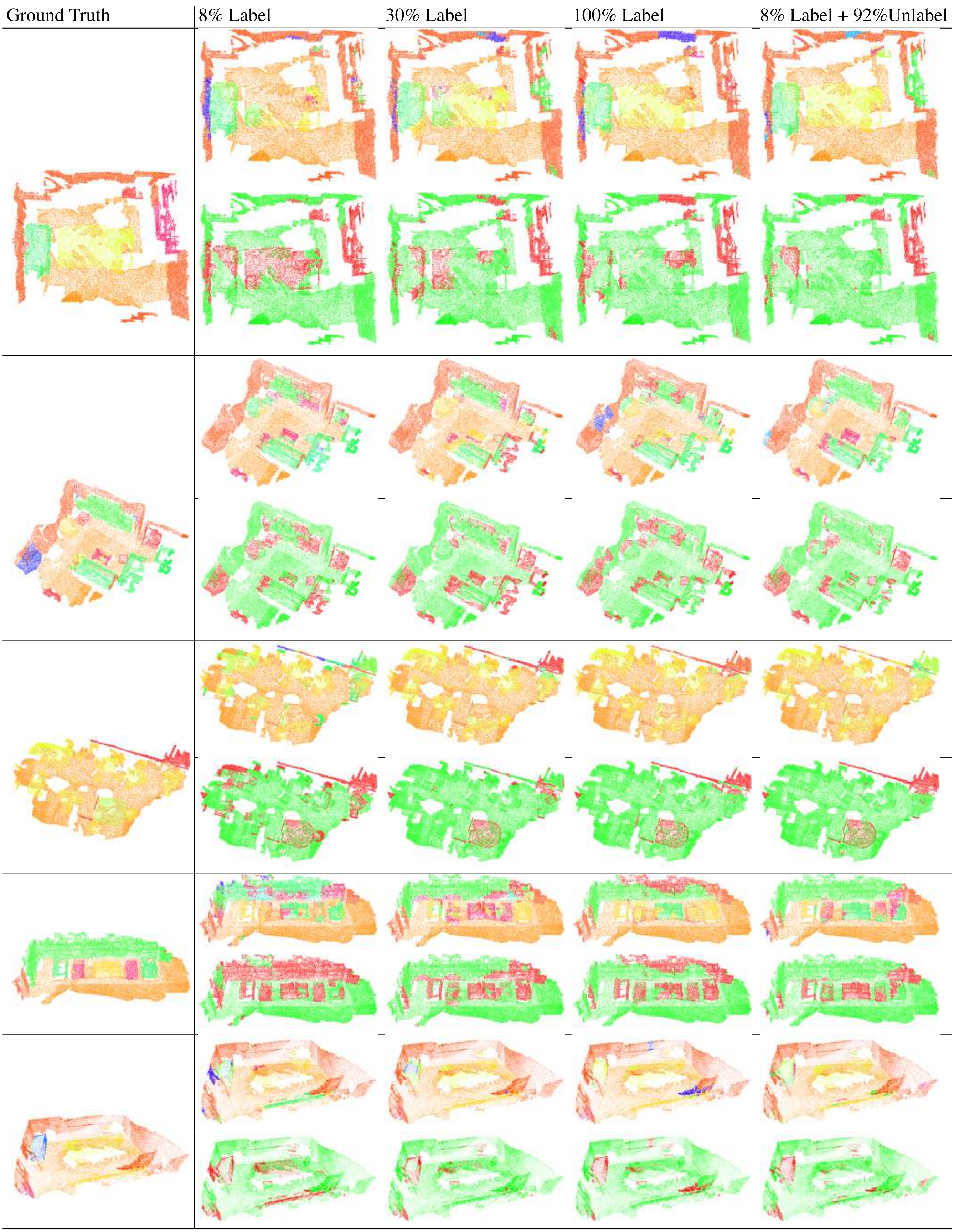
D.2. Annotated Feature Points

D.2.1 SHREC07

We used annotated feature points provided by [26]. The number of key points per category range from 11 (e.g. Plane) to 36 (Human).

D.2.2 ShapeCoSeg

Note that the models in ShapeCoSeg [50] are originally associated with annotations of semantic segments. Such annotations, however, are not ideal for establishing dense correspondences. To address this issue, we employed AMT to annotate semantic feature correspondences across the entire dataset. Note that in some cases, the feature correspondences are not purely based on 1-1 correspondences (e.g., multiple handles). When performing experimental evaluation, we evaluate the geodesic error to the closest feature point of the same type for experimental evaluation.


Ground Truth
8% Label
30% Label
100% Label
8% Label + 92%Unlabel