7. Supplementary Material

Supplementary material includes

- Order Invariant Joint Matching Algorithm
- Implementation detail
- Additional Qualitative results
- Additional baeslines and additional numerical comparisons

7.1. Order Invariant Joint Matching Algorithm

Algorithm 1 Order Invariant Joint Matching Algorithm

Sort all the nodes in the connectivity graph by the degree of the target node in descending order.
while not all the nodes are assigned **do**
    Start with the most connective node as the target node i and assign it with sign *peg*.
    for all neighbor node j of the target node **do**
        if j has been assigned with *peg* **then**
            Add (i, j) pair to conflict cache
        else
            Assign *hole* to neighbor node j
        end if
    end for
end while
for all edges e in the connectivity graph **do**
    if the two nodes of e have the same label **then**
        Add the edge to conflict cache
    end if
end for
for all edge=(i,j) in conflict cache **do**
    if one of $\{i, j\}$ is congruent **then**
        Assign congruent part with *peg*
    else
        Remove (i, j) from connectivity graph
    end if
end for

The permutation invariance nature of the losses schemes forbids us to use the ground truth matched pairs, as there lack of a global rule in the ground truth joint matching annotation that defines what should be the peg and what should be the hole. For example, for four chair legs that are connected to the chair seat, a set of the leg joints are defined as pegs and the other ones are defined as holes and vice versa for their mated seat joint. In the Figure 5, the four legs are geometrically interchangeable. However, the ground truth joint matching annotation defines the front right leg, part 2, and its corresponding seat joint are connected via leg-peg and seat-hole joint pair. The front left leg, part 4, and its corresponding joint is connected via leg-hole seat-peg joint pair. These ground truth matching scheme prevents the chair legs to be interchangeable if the network predict the reversed placement for the two legs, as the matching rule defines that only different type of joint can be mated. For example, if part 2 is now in place of part 4, which means that the leg-peg and seat-peg are now placed next to each other.

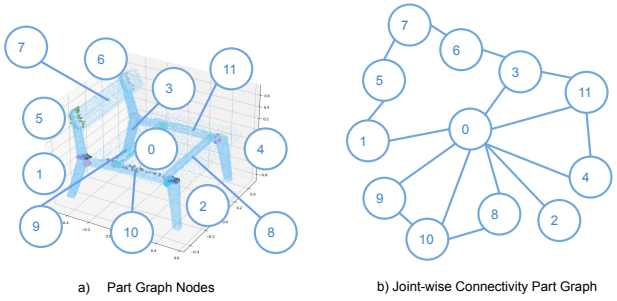

Figure 5. Example joint-wise connectivity graph for a ShapeNet Chair

A second problem is with part joints that are very close to each other on the same part for example the back legs of the example chair in Figure. 5 has three joints in the same area, and all of them needs to be the same type, so that there is a signal that tells the network that they cannot be jointed, because otherwise these joints that are always very close to each other regardless of the part pose prediction. Yet, simply defining joints from the same part to be of the same sex does not do the job, as it will fail when there are loops in the joint graph, especially with odd element loops. For example, part 0, part 9, and part 10. If part 0 is defined to be a peg part, then part 9 and part 10 will simultaneously become hole parts. However, part 9 and part 10 are connected to each other, thus this simple scheme does not work in such cases. We introduce the Order-Invariant Joint Matching Algorithm that first leverage graph traversal to assign joint-type in a part-wise manner then deals with the conflicted assignment separately, as shown in Alg. 1.

7.2. Implementation Detail

Our Method. We organize our input joint peg-hole points $\mathcal{J} = \{j_i^{p,h}\}_{i=1}^{M}$ as ternary mask $\{m_i^{p,h}\}_{i=1}^{M}, m_i^{p,h} \in \{-1, 0, +1\}$ for the part point clouds $\mathcal{P} = \{p_i\}_{i=1}^{N}, p_i \in \mathbb{R}^3$, where the mask values of -1 denotes the peg points, $+1$ for hole points, and 0 for non-joint point. Applying each joint mask to its parent part point cloud $\mathcal{J}^p = \{p_i^{joint} = (p_i; m_i^{p,h})\}_{i=1}^{2M}$ forms 4-dimensional point cloud containing information of both joint points and geometry of the part $\mathcal{P}^j = \cup_{t=1}^{n_i}\{p_t^{joint}\}$. All message-passing modules $f_{v\xrightarrow{e}}, f_{e\xrightarrow{v}},$, relationship reasoning modules $f_{v\xrightarrow{r}}, f_{r\xrightarrow{v}}$,and pose prediction modules f_{pose} are modled using Multi-Layer Perceptron (MLP) with hidden dimension of 128. We predict part poses $q \in \mathbb{R}^7$, composed of a unit 4-dimensional quaternion rotation vector and a 3-dimensional translation vector. A design choice was made for $\delta q_{i,t}$ such that translation $\delta t_{i,t}$ is cliped at a small range between $[0.0, 0.01]$. Empir-

ically, setting $\delta R_{i,t}$ to sample from a set of flip directions, *e.g.* $\pi/4, \pi/2, ...$ improves performance.

Following [26], we define $\mathcal{L}_{shape}$ as a weighted combination of local part translation loss $\mathcal{L}_t$, rotation loss $\mathcal{L}_r$ and the global shape structure loss $\mathcal{L}_s$. Empirical evidence has shown that parameter values of $\{\lambda_1 = 1.0, \lambda_2 = 10.0, \lambda_3 = 1.0\}$ produce the best results. We define our joint loss $\mathcal{L}_{joint}$ to be a weighted combination of L2 distance between the means of the matched joints $\mathcal{L}_{jL2}$, Chamfer distance of matched joint points $\mathcal{L}_{jcd}$ as well as local part L2 rotation loss $\mathcal{L}_{rL2}$. Empirical evidence has shown that parameter values of $\{\lambda_4 = 1.0, \lambda_5 = 5.0, \lambda_6 = 1.0\}$ produce the best results.

Baselines. We follow the same implementation scheme used in [12, 15] for B-Complement, B-LSTM, B-Global, B-GNN, using the official code release provided by Huang et al. [15]. For B-NSM, we use the official code release by Chen et al. [7], and we modify the attention module in transformer. As the original setting only considers two parts, and the original attention module used in [7] compares the feature vectors of the two parts. Since we tackle a multi-part multi-joint setting, where different shapes contain various number of parts and different parts contain various number of joints. Directly applying pair-wise attention on all possible pairs between all parts and all joints is computationally inefficient. Therefore, we encourage implicit relationship reasoning using self-attention by applying the same attention module on feature vectors compressing all parts and joints information of a given part set.

B-Joinable modifies the original method [48] to take point cloud data of multiple parts. We preserve the geometry encoding and joint-axis prediction philosophy of the original method. We encode part point cloud data and predict a 3 DOF axis for each matched joint. The predicted joint axis is used to further predict the 6 DOF poses for each part. The contact B-Rep joint loss functions proposed by Willis et al. has similar objectives to our joint losses for point clouds. In addition to the setting of imposing our loss functions, we also experimented with supervising for joint axis. We add additional supervision signal calculated from contact joints normals. Empirically, we discover this additional supervision signal hurts performance. Because without the correct pairing of joints, the joint axis signal add noise to the supervision. In our paper, we report the best performing scenario for our task.

We train and test all baselines and our method using a work station equipped with 16-core AMD CPU, and 2 NVIDIA 3090 GPU. All methods are trained to full convergence for 18 hours. More details on our data and method can be found here: https://github.com/AntheaLi/multi-joint-assembly

7.3. Additional Qualitative Results

Additional qualitative examples shown in Figure 6.

7.4. Additional Baseline Comparisons

We also compare our work with the suggested [51] and [44]. Wu et al. [50] focuses on part equivariant simple shape (8 parts max) asembly and does not look at joints. We repurpose the method to tackle our task . SPSM [44] tackles a different task. It is a generative model that assumes known connectivity graph to generate diverse shapes, whereas we try to solve for a connectivity graph for a given set of parts. Reformulating [B] to tackle our task is non-trivial, we use their proposed sequencing then optimizing scheme. We show the comparison results of B-VNN and B-SPSM with our method in Table. 4 Our task data uses more complex shapes of 20 parts and 50 joints

In addition to these two baseline methods, we also provide the baselines' performance with our joint-aware part input but use their original losses, shown in Table 4. We notice that the baselines perform the worst under this setting. We think that adding the joint input does not help the baselines to implicitly infer structured connectivity. Instead, without explicitly joint-centric loss design, the joint information is treated as noise by the baseline methods. This results in decreased baseline performance. Additionally, we also notice that baseline dynamic [15] exhibit the lowest performance on the Cabinet category, and cannot infer reasonable pose predictions.The Cabinet category is not included in its original experiments.

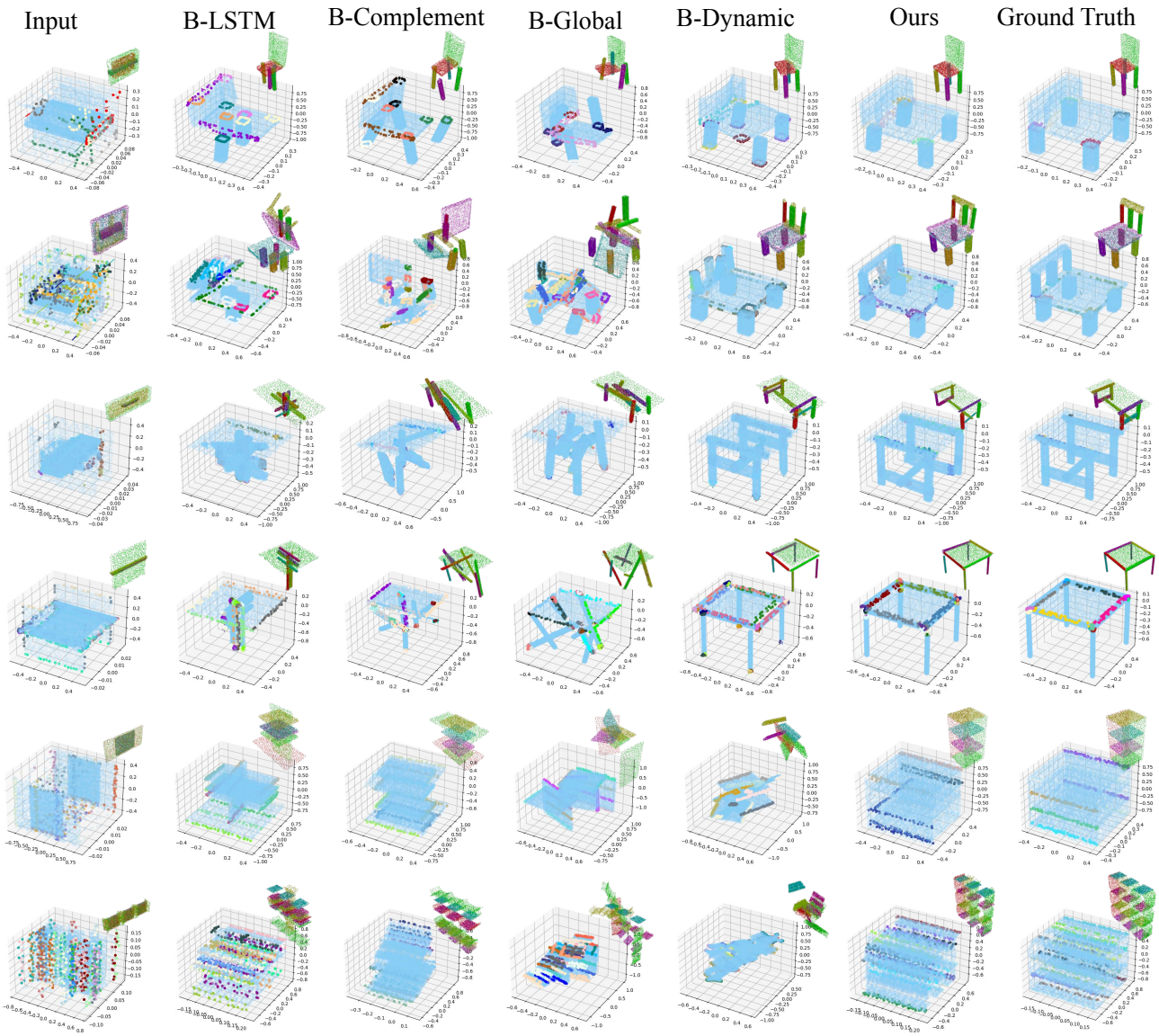

Figure 6. Qualitative comparison of our method and baselines in their best performing setting, their original setting.

		Shape Chamfer Distance ↓			Part Accuracy ↑			Joint Chamfer Distance ↓			Joint Accuracy ↑		
Setting	Method	Chair	Table	Cabinet	Chair	Table	Cabinet	Chair	Table	Cabinet	Chair	Table	Cabinet
Original Setting	B-Global	0.015	0.013	0.008	32.8	30.1	33.6	0.712	0.847	0.667	13.4	15.8	10.7
	B-LSTM	0.017	0.026	0.007	39.4	22.5	44.4	0.756	0.728	0.651	17.0	13.2	14.8
	B-Complement	0.028	0.034	0.222	11.0	5.33	0.0	0.901	0.977	1.074	7.54	8.30	23.6
	B-VNN	0.468	0.629	0.470	14.1	11.0	7.7	0.618	0.711	0.691	12.1	11.6	13.2
	B-GNN	0.007	0.008	0.006	65.3	61.4	45.0	0.725	0.855	0.683	24.4	30.0	18.6
Our Input Setting (joint input)	B-Global	0.076	0.031	0.028	3.9	12.4	5.7	1.724	1.239	1.281	3.2	4.8	2.7
	B-LSTM	0.028	0.039	0.010	7.3	3.8	27.2	0.777	0.708	0.661	8.7	15.1	6.7
	B-Complement	0.055	0.047	0.015	2.9	7.2	15.8	1.413	1.004	0.767	5.9	6.5	10.1
	B-GNN	0.046	0.076	0.035	4.3	2.1	3.8	0.547	1.120	0.639	15.4	8.1	7.8
Our Full Setting (joint input and loss)	B-Global	0.029	0.022	0.013	5.4	12.0	15.0	0.513	1.268	0.488	12.7	4.0	6.9
	B-LSTM	0.037	0.029	0.017	4.4	4.1	15.3	0.394	0.875	0.467	20.3	7.7	13.8
	B-Complement	0.048	0.044	0.029	4.5	8.0	11.6	0.456	0.647	0.503	17.2	15.5	17.0
	B-SPSM	0.056	0.055	0.037	4.1	6.4	9.7	0.601	0.624	0.682	18.3	13.6	21.4
	B-VNN	0.023	0.025	0.016	4.2	3.9	9.4	0.468	0.629	0.470	14.1	11.0	7.7
	B-GNN	0.034	0.039	0.021	11.5	3.2	10.4	0.379	0.786	**0.416**	21.5	10.3	20.0
	Ours	**0.006**	**0.007**	**0.005**	**72.8**	**67.4**	**63.3**	**0.352**	**0.602**	0.620	**57.2**	**50.6**	**27.5**

Table 4. Quantitative comparison between our approach and the baseline methods.