Supplemental Material for Global Registration of Dynamic Range Scans for Articulated Model Reconstruction

WILL CHANG
University of California, San Diego
and
MATTHIAS ZWICKER
University of Bern

1. PIVOT AND SLIDING JOINTS

Although our system currently supports ball and hinge joints, it can be easily extended to handle pivot and sliding joints as well. A pivot joint is one that allows 1 DOF rotation about a line, similar to the hinge joint. Sliding joints allow 1 DOF translational movement about a line. For completeness, we discuss how to extend our method to handle these joints.

Since both hinge and pivot joints are rotations about a fixed axis, our mechanism for detecting and constraining hinge joints already works for pivot joints as well. However, sliding joints are not currently handled by our system. To detect a sliding joint, we check if the the relative transformation between two neighboring parts is a translation along a common line. The relative transformation between pair (a, b) for frame i can be computed by first transforming using a to the reference frame, then applying the the inverse of b:

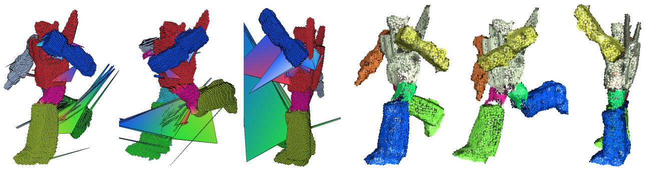
$$T^{(i)}_{a\to b} = \left[R^{(i)}_{a\to b}, \vec{\mathbf{t}}^{(i)}_{a\to b}\right] = T_b^{(i\to\text{Ref})^{-1}} \circ T_a^{(i\to\text{Ref})} = \left[R_b^{(i\to\text{Ref})^{\top}} R_a^{(i\to\text{Ref})},\ R_b^{(i\to\text{Ref})^{\top}} \left(\vec{\mathbf{t}}_a^{(i\to\text{Ref})} - \vec{\mathbf{t}}_b^{(i\to\text{Ref})}\right)\right].$$

To check if this is a translation, we can threshold the maximum deviation of each relative rotation $R^{(i)}_{a\to b}$ from identity. If this test passes for all frames i, then we can check if the translations are along a common line by fitting a line to the translations. To do this, we can perform PCA on the set of the relative translations $\vec{\mathbf{t}}^{(i)}_{a\to b}$ using the SVD. If the distribution of translations is maximal along the first principal component and minimal along the other two, then the translations are well-described by the line that follows the direction of the first principal component. This can be determined quantitatively by checking if the ratio of the sum of the two smaller singular values to the sum of all singular values is less than a threshold. The final parameters for the sliding joint's axis is given by the the line that points in the direction of the first principal component and passes through the mean of the relative translations.

To constrain a sliding joint, we can use a set of points $\mathbf{p}_{ab}$ generated around the sliding axis (in the reference frame) to constrain the motion. For the points we can take the vertices of a cylinder (generated with a small number of slices and stacks) centered on and aligned to the sliding axis. To constrain the motion, we add the following equation to the objective:

$$\left\| \left(T_a^{(i\to\text{Ref})^{-1}}(\mathbf{p}_{ab}) - T_b^{(i\to\text{Ref})^{-1}}(\mathbf{p}_{ab})\right) - R_a^{(i\to\text{Ref})^{-1}}(\vec{\mathbf{v}}_{ab}) \left[\left(T_a^{(i\to\text{Ref})^{-1}}(\mathbf{p}_{ab}) - T_b^{(i\to\text{Ref})^{-1}}(\mathbf{p}_{ab})\right) \cdot R_a^{(i\to\text{Ref})^{-1}}(\vec{\mathbf{v}}_{ab})\right] \right\|^2,$$

where $\vec{\mathbf{v}}_{ab}$ is the direction vector of the sliding axis in the reference frame. This objective specifies that, when $\mathbf{p}_{ab}$ is transformed to frame i using transformations a and b, the resulting distance stays the same when projected to the sliding axis.



(a) Result Using a Grid-Based Representation (144.00 sec/frame)

(b) Result Using a Graph-Based Representation (28.36 sec/frame)

Fig. 1. Comparing registration results using grid-based and graph-based weight representations. These images show the represented weight function deformed into different poses according to the optimized transformations and weights. The artifacts with the grid are absent when using the graph.

2. GRID-BASED WEIGHTS VS. GRAPH-BASED WEIGHTS

We compare the benefit of using a graph vs. using a grid for representing the weight function. We implemented the simultaneous registration using a grid and compared the results to a graph-based implementation. First, we found that the performance of the graph-based registration is much faster, because the grid-based method has an additional overhead of translating the weights from the grid to the samples. For processing the 90 frame robot sequence, the global registration took a total of 144.00 seconds per frame using the grid strategy, while it only took 28.36 seconds per frame for the graph based strategy (excluding initialization time in both cases).

Second, the graph-based representation dealt robustly with topology issues. Since we can prune edges of the graph based on the optimized motion, the algorithm can handle topologically difficult cases robustly. An example of this is shown in Figure 1, where we display the grid and graph deformed according to the optimized weights and transformations. The grid based result shown on the left has many problems where grid cells stretch apart. This is because the limited resolution of the grid cannot resolve the left and right leg of the robot when they move close to each other. In contrast, the graph-based result shown on the right does not suffer from this issue.

3. COMPARING WITH REFERENCE MOTION

To validate our method, we compared our reconstruction results with the original (i.e. ground truth) mesh and animation data that was used to generate the synthetic walkman dataset. As a supplement to the graphs shown in Figure 14 of the paper, we show a comparison plot of the numerical values of the transformations for

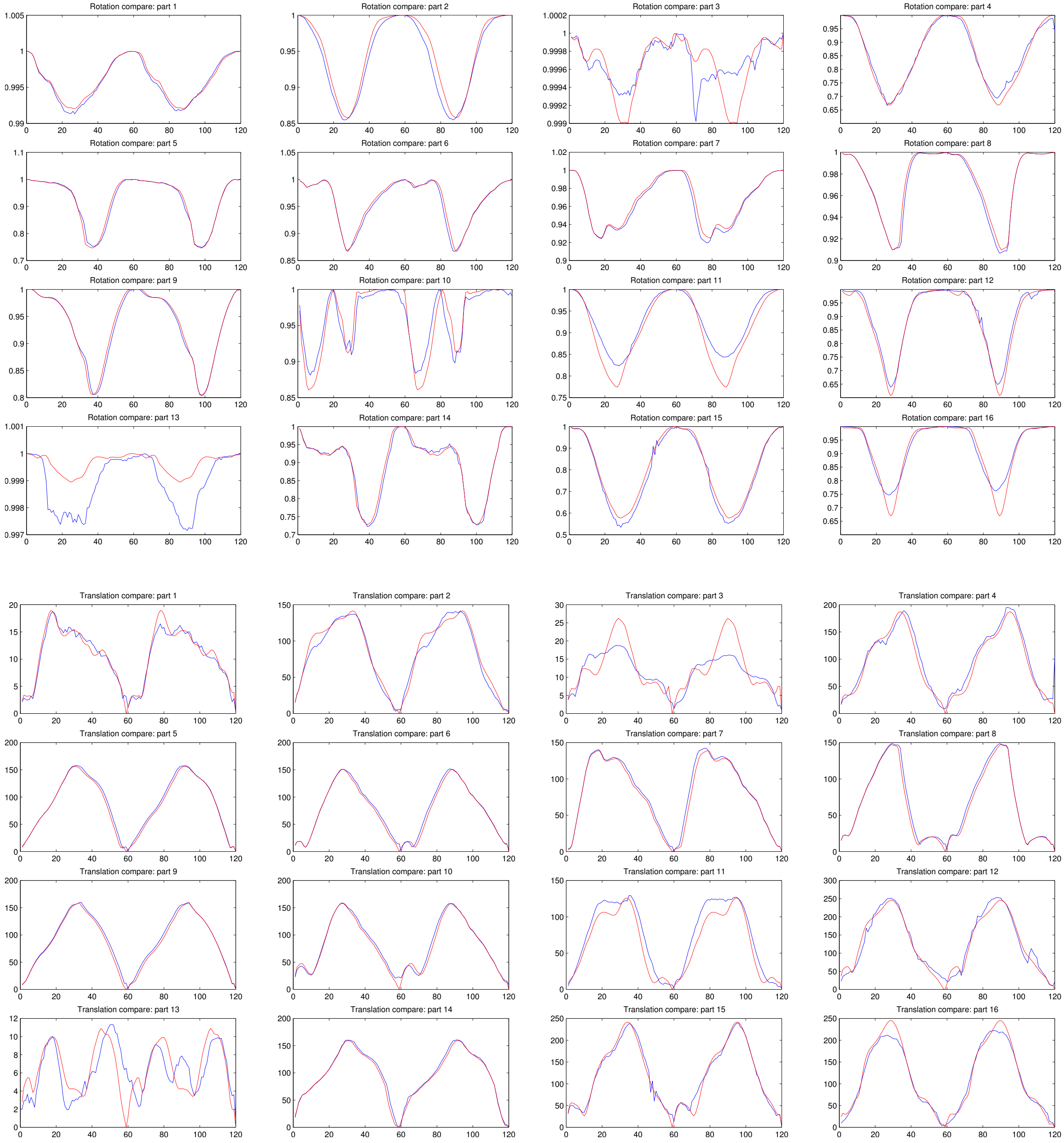


Fig. 2. Comparing the estimated transformation parameters for each part with the reference parameters used to generate the dataset.

each part of the skeleton in Figure 2. We compare a total of 16 parts. The top 16 graphs (top 4 rows) show a comparison of the rotation angle vs. time, and the bottom 16 graphs (bottom 4 rows) show a comparison of the translation vector vs. time. Red is used for the reference (ground-truth) data, and blue for the reconstructed data.

4. PARAMETER COMPARISON TESTS

We show the results of additional experiments performed to analyze the effect of different parameter choices.

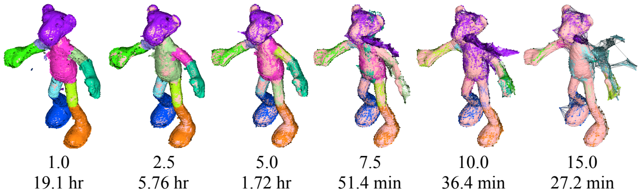
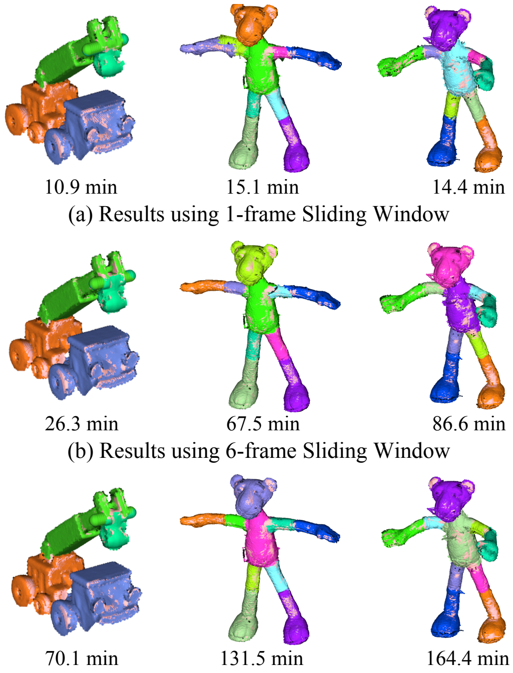


Fig. 3. The sample spacing size affects the registration. The two numbers beneath the images denote the sample spacing size (top) and the computation time (bottom).



(a) Results using 1-frame Sliding Window

(b) Results using 6-frame Sliding Window

(c) Results using 12-frame Sliding Window

Fig. 4. The sliding window does not affect the registration quality heavily.

4.1 Effect of the Sample Set Size

The size of the DSG depends on the sample spacing. A small sample spacing results in a dense DSG, whereas a large sample spacing results in a sparse DSG. Figure 3 shows the result of processing the pink panther dataset with a varying sample spacing value.

We see that the smaller sampler spacings tend to produce good registrations, and the registration may fail if the sample spacing is too large. However, denser DSGs require more memory and computation time. Generally denser DSGs are needed as the number of articulated parts increase. However, you don't need to have too dense a DSG, as long as there are enough points to distinguish each part. Denser DSGs also tend to produce more accurate registrations, because there are more points to evaluate the distance error so that we can capture fine-scale errors.

4.2 Effect of the Sliding Window Size

We perform an experiment where we vary the size of the sliding window (Figure 4). We found that the registration quality does not depend much on the sliding window size. Even a sliding window size of 1 frame is enough.

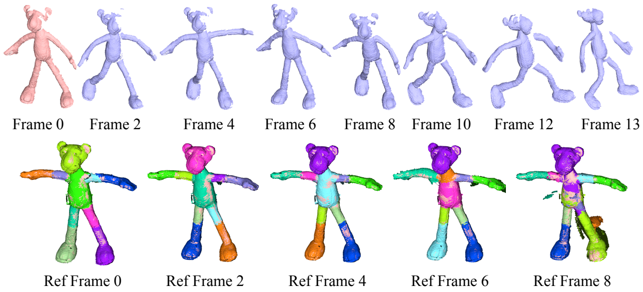


Fig. 5. Results when aligning the pink panther dataset with different reference frames. The top row shows some of the original input frames, and the bottom row shows the registration result using different reference frames.

Table I. Performance numbers for reconstructing 62 frames of the walkman dataset, scanned using varying resolutions.

Resolution	Points	Time(sec)	Mem(MB)	Sec/frame	Points/frame	Msec/pt
240x180	93765	2333.8	60.1	37.6	1512.3	24.9
400x300	474636	12093.2	229.0	195.1	7655.4	25.5
520x390	842932	16859.9	400.5	271.9	13595.7	20.0
600x450	1146627	24054.1	548.2	388.0	18494.0	21.0

Larger sliding windows give more “second chances” to refine the registration in case the articulated structure changes in a later frame. What really matters in the global registration is that the articulated structure (weights) is estimated globally over all frames. This is the main reason why the sequential method fails in Figure 16. Accumulating the point cloud and throwing away the scan discards the motion information in previous frames and results in an inaccurate registration.

4.3 Effect of Reference Frame Choice

The algorithm can fail with different reference frame choices. It works best when all parts are visible for the reference frame and several frames that follow. Also, the registration is most successful when all joints are exercised early in the sequence. Figure 5 shows results when aligning the pink panther dataset with different reference frames (earlier frames are discarded). The registration is less successful in the rightmost example, because the algorithm encounters frames with occluded surface parts earlier in the optimization.

4.4 Performance Impact of Mesh Size

We run a scan simulation of the walkman dataset with varying scan resolutions. This gives geometric data, but with varying mesh size (scanned point density). We reconstruct the model using each resolution and summarize the performance in Table I. The total computation time and memory usage increases, but the time spent per point stays approximately the same.

4.5 Memory Requirements

Our implementation stores all scans in memory, maintains extra data for each scan (normals, boundary information, sample set, k-d tree for computing closest points, transformations for each part, precomputed distance error, occlusion information) and for saving the global registration state (DSG samples and graph structure, weights for each sample). We measured that typically we need about 400 MB of memory for 90 frames, with 60 MB for the range scan data. This is highly variable and depends on the range scan data size and the DSG sample spacing.