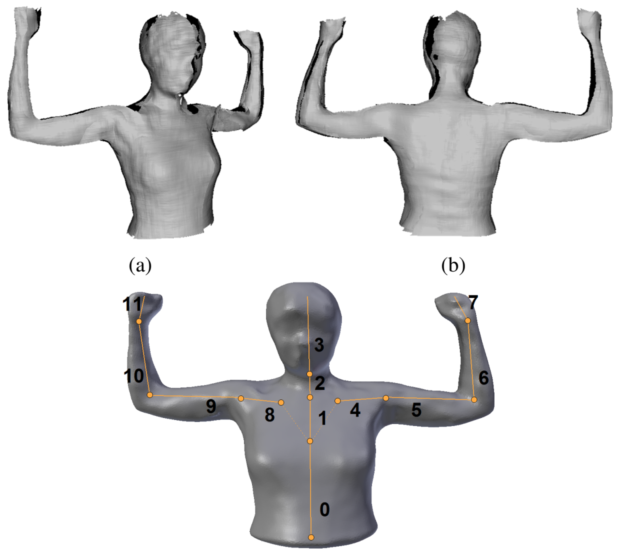
Markerless Motion Capture with Multi-view Structured Light

Ricardo R. Garcia, Avideh Zakhor; UC Berkeley; Berkeley, CA



(a) (b)

(c)

Figure 1: Using captured partial scans of (a) front and (b) back to generate a (c) closed template generated by Poisson surface reconstruction. The bones and joints are numbered and indicated in orange.

Abstract

We present a multi-view structured light system for markerless motion capture of human subjects. In contrast to existing approaches that use multiple camera streams or single depth sensors, we reconstruct the scene by combining six partial 3D scans generated from three structured light stations surrounding the subject. We avoid interference between multiple projectors through time multiplexing and synchronization across all cameras and projectors. We generate point clouds from each station, convert them to partial surfaces, and merge them into a single coordinate frame. We develop algorithms to reconstruct dynamic geometry using a template generated by the system itself. Specifically, we deform the template to each frame of the captured geometry by iteratively aligning each bone of the rigged template. We show the effectiveness of our system for a 50-second sequence of a moving human subject.

Introduction

Human motion capture has been under investigation for many years in different applications [16, 17]. While marker based motion capture has been popular for a while, it is desirable to do away with markers, simplify the capture process, and improve realism by capturing actual subject color and shape.

In many systems, the geometry of the dynamic human subject is captured using multi-camera setups [22, 19, 4, 23, 6]. These systems recover a representation of the human subject by using visual hulls generated from each camera view. Many approaches use a template of the human model to assist in the reconstruction of the dynamic geometry [4, 23, 6, 3, 5]. The methods using templates can either use a skeleton [23, 6, 3] or not [4, 5]. In [23], Vlasic et al. generate a detailed template using a high quality laser scanner. The template is then deformed according to constraints gathered from the observing cameras. De Aguiar et al. similarly present a template based method that starts with a high quality laser scan, but do not use a skeleton to deform the mesh [4]. Rather, mesh based deformation techniques such as those in [20] are used to deform the mesh according to the visual hulls of the observing cameras. The method in [23] fits a template via a combination of coarse movements from skeleton deformation, followed by local mesh deformation to capture the details of the scan. Gall et al. use a similar approach, but do not require manual user intervention [6]. Methods using highly detailed templates are sometimes criticized for "baking in" details that should not be present throughout the reconstructed sequence.

Besides multi-camera methods emphasizing the use of visual hulls, other capture methods have been proposed. In [24], dynamic scans from a photometric stereo system are used for surface reconstruction. While the geometry captured from this system is quite accurate and detailed, the system is complex and expensive. Unlike methods involving visual hulls from multiple-cameras, in [24], the 3D shape of the human subject's surface is directly captured. Such systems directly reconstruct the subject's surface without any prior information rather than using a template to estimate the pose and shape of a human subject. For instance, the approach in [24, 13] does not use templates to reconstruct water tight meshes. The results from these direct capture approaches often yield errors in the topology of the reconstructed geometry since there is no prior information on the shape of the model. Similar works have focused on the problem of registering and generating consistent geometry from sets of dynamic point clouds [15, 21, 18]. These methods only work with high temporal sampling and limited occluding geometry.

Other methods approach the problem of motion capture by fitting data-driven models to the captured subject [1, 2]. In [1], the motion of a human subject is captured using motion capture markers coupled with data-driven models representing human shape and pose. Using a combination of rigid and non-rigid deformation, their method is able to capture the geometry and motion of human subjects. Key to this approach is the acquisition of high quality body scans of many body types in many poses. In [14, 26], the dynamic reconstruction of a human subject is accomplished using a single depth sensor. In [26], a pre-processing step of scanning the performer in multiple poses is used, and a custom template is generated and rigged for later motion capture.

In this paper, we present a custom markerless motion capture system consisting of multiple structured-light subsystems to capture geometry of a human subject from surrounding views. By precisely controlling the synchronization of all cameras and projectors in our system, we enable interference-free capture from multiple structured-light scanners in a single system. As with any multi-view system, since the entire shape of the human subject is not always visible from any one capture device, we opt to use a template to represent the shape and pose of the human subject. However, unlike existing approaches which require additional scanning hardware or complex template initialization processes, our system is able to directly generate a custom template for each scanned human subject. Specifically, from an initial scan of the human subject in an occlusion free pose, we create a customized template for that subject as shown in Fig. 1b. Finally, we propose a template deformation method that directly tries to fit a rigged skeleton to partial geometry scans while obeying specified joint constraints. The problem of fitting a template to the human subject is greatly simplified if constrained by 360-degree geometry from three stations.

There are existing methods for reconstructing dynamic geometry from multiple 3D geometry scans. Dou et al.[5] use eight Kinects surrounding a human subject to build a model of a dynamic figure. Due to sensor noise, they cannot directly create a template in a single frame; rather, they learn it over time. Our system is most similar to [9] which captures two opposing views of the human subject and stitches the views together, leaving significant artifacts around the seams.

The outline of this paper is as follows. First, we present the specifics of the system architecture and hardware used to capture 3D partial reconstructions of the human subject. The next section describes how we generate and deform the template to fit partially captured geometry scans. The final two sections present an example of processing real captured human motion and conclusions, respectively.

System Setup and Data Capture

Our proposed system consists of three distinct structured-light stations, shown in Fig. 2, surrounding a central capture volume with the devices located on each station pointing towards the center of the capture volume. Each station is equipped with one DLP® projector, two grayscale cameras, and one color camera in a configuration similar to [25]. To prevent interference between the three projectors at each structured-light station, each station takes turns illuminating the human subject in a time-multiplexed sequence. The system is controlled by two separate desktop PCs, one for driving structured-light patterns to the three projectors, and a second one for streaming in images from all nine cameras. A microcontroller is used to extract timing signals from the projectors to generate trigger signals for all of the cameras. An external PCIe expansion chassis is added to the camera PC to provide enough Firewire cards to connect all nine cameras to the single PC. The resolution of the projectors is 1024×768, and the resolution of the cameras is 640×480. At each station, the two grayscale cameras are used with the local projector to capture geometry, and the color camera is used to capture texture. As described in [27], the projectors are modified to only project in grayscale by removing the color wheels. DLP® projectors generate color images by sequentially projecting the individual red, green, and blue color channels of a video source. The removal of the color wheel effectively modifies the projector from a 60 Hz color projector to a 180 Hz grayscale projector. Similar to [27], we use three phase-shifted sinusoidal patterns to reconstruct depth. The three sinusoidal patterns are individually captured as they illuminate the human subject and the phase of each pixel in the camera is computed [27]. While the phase of a camera pixel determines which points in the projector potentially illuminate it, due to the periodic nature of the sinusoidal patterns, correspondences between the cameras and projectors are not uniquely known by the phase alone. By using the temporal phase unwrapping method from [7], the phase images can be unwrapped over time and space to ensure spatiotemporally consistent point clouds at each station.

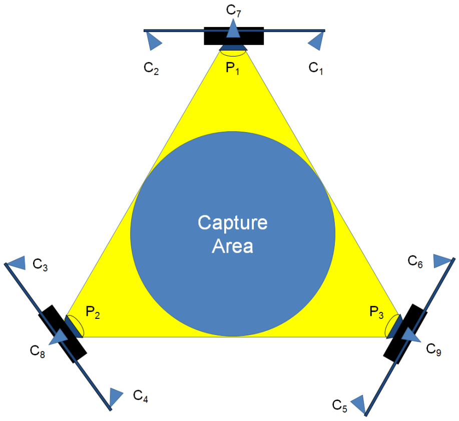


Figure 2: Top-down view of the three structured-light stations in the system. Geometry cameras are labeled C_1-C_6. Color cameras are C_7-C_9. Projectors are P_1-P_3.

If multiple projectors simultaneously illuminate the same point on the human subject, neither the phases of projectors can be accurately estimated, nor can the correspondence between camera and projector pixels. Since the projectors natively operate at a 60 Hz rate, the time-multiplexing between the three stations lowers the capture rate from each station to 20 frames per second. To implement this interleaving process, the video signals sent to each projector are all synchronized to each other. Specifically, we use two NVIDIA® Quadro® 5000 graphics cards with an additional G-Sync® daughter card to synchronize all video source signals. Triggering signals sent to the capture cameras are derived from the synchronized projectors. Similar to [27], the grayscale cameras capture the individual structured-light patterns at approximately 180 Hz, and the color cameras capture at 60 Hz. The exposure of the color cameras is chosen to last for the entire duration of projection of the three sinusoidal patterns; thus the integration of the three phase-shifted sinusoidal patterns appears as a constant illumination to the color cameras. All camera and projectors are calibrated using the approach in [8].

Even though the evolving pose of the human subject can be observed frame by frame after point cloud alignment, the limited number of capture devices and stations surrounding the human subject causes holes during reconstruction. As shown in Fig. 2, there are several holes in each of the views which need to be filled out in the fused 3D model. This motivates the use of templates in our system.

Template

Even though visual hulls can be used to estimate geometry for many poses of the human subject, for scenes with significant self-occlusions, it is not always possible to generate a watertight representation of the entire subject's surface. As such, *a priori* information on the structure of the human subject can help to properly fill in these regions. We use a template to create a complete representation of each frame. However, unlike existing methods [4], we use the same system configuration that captures the dynamic geometry to also capture the template, removing the need for a separate system such as a high-quality laser scanner.

To create a template, the human subject is captured in a pose where the majority of the his or her surface can be observed by the cameras and projectors so as to limit self-occlusions. Figs. 1a and 1b show the captured geometry with the human subject in the template pose. The majority of the geometry is captured, except for regions on the tops and bottoms of the arms, the top of the head, and the bottom of the torso, which are not observable by any of the capture devices. We use Poisson surface reconstruction [11] to fit a smooth meshed point cloud over the existing geometry. As seen in Fig. 1c, this generates a hole-free representative template of the human subject.

Even though the template provides an *a priori* reference of the true shape of the human subject, as a mesh alone, it lacks the structure to effectively fit the fused captured geometry from the three stations. In poses with significant occlusions, a reference to the overall structure of the template is useful to prevent unnatural mesh deformations. As is done in traditional motion capture, we fit a skeleton rig to the template to control the way its geometry is deformed over time [16].

The skeleton rigging, or skeleton, emulates natural pose deformations of the template mesh. The position and movement of the bones control the movement of the vertices in the template mesh. For our system, which only captures a waist-up view of the human subject, we use 12 bones within our skeleton, as shown in Fig. 1c. [1] Each bone in the skeleton has a single lead joint that controls its movement and position. The joint-bone pairs are labeled in Fig. 1c. The position of each vertex in the template mesh is determined by the positions and orientations of nearby bones. The influence of each bone on each vertex's position is specified by the bone weight map matrix $\mathbf{W}$ with dimensions of $[N \times J]$, where N is the number of vertices in the template mesh and J is the number of bones in the template. Each entry in the matrix $\mathbf{W}(i,j)$ takes on a real value from zero to one, specifying the influence of each bone j on the position of each vertex i by taking into account a vertex's relative distance to each bone.

We deform the template by using dual quaternion skinning [10] to modify the skeleton pose so as to fit the fused geometry from three stations. We solve for the bone positions that deform our mesh to best align the template to the fused surfaces captured in each frame.

As is standard with skeleton-rigged deformation, rotating a branch around a joint not only rotates the connected bone, but also all subsequently connected bones. Specifically, the skeleton of our template is connected in a tree structure with joints serving as the nodes of the tree and the bones as the paths between nodes. We can think of the set of bones that follow from a joint in the skeleton to a terminal bone in the tree as a branch. The largest branch starts from the joint for bone 0 in Fig. 1c and includes all bones in the skeleton. Anytime a bone is transformed around its joint, the subsequent bones along the branch need to be updated as well. We refer to the branch that starts at a bone j as branch j. A branch weight map $\bar{\mathbf{W}}$ can be calculated from the bone weight map:

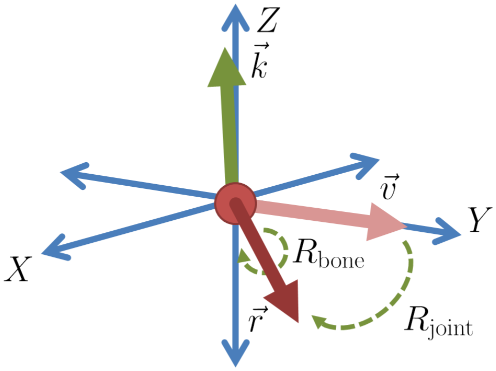


Figure 3: An example of the decomposition of $\mathbf{R}_{\text{local}}$ into $\mathbf{R}_{\text{joint}}$ and $\mathbf{R}_{\text{bone}}$. The original pose of a bone in P_0 always lies along the y-axis with unit vector $\vec{v}$. The axis of rotation used to rotate between $\vec{v}$ and final bone position $\vec{r}$ is $\vec{k}$.

$$\bar{\mathbf{W}}(i,b) = \sum_{j \in B(b)} \mathbf{W}(i,j), \tag{1}$$

where b is the current branch and $B(b)$ is the set of bones in a branch.

For each bone, we specify a range of motion under which it can be rotated relative to its parent bone. A parent bone p_j is defined as the bone connected to bone j that is higher in the skeleton tree and connected to the lead joint of bone j. For example, in Fig. 1c, bone 5 is the parent of bone 6. We define a local right-handed coordinate system for each bone in the original pose P_0 of the template, shown in Fig. 1c, that places each joint at the origin and the bone out of that joint on the y-axis as shown in Fig. 3. The constraint set for each joint is generated by inspecting each joint in the template under varying amounts of rotation and selecting a rotation range for the joint relative to its parent that falls within the physical range of motion for a human subject. Specifically, the constraints set for each joint are defined as minimum and maximum limits of rotation on each of the x, y, and z-axes relative to its parent joint. The rotation constraints can be applied directly only when a joint rotates about a single coordinate axis. Since in practice, this is not often the case, we need to compute a different constraint representation based on the values from the individual axis rotation constraints. To do this, we represent the constraint for each joint in two parts. The first component $\mathbf{R}_{\text{joint}}$ is a rotation matrix that represents the portion of the joint rotation that occurs around an axis lying in the x-z plane of the joint's local coordinate system as shown in Fig. 3. This rotation moves the bone from the y-axis to its new position shown in red in Fig. 3. The second component of the joint rotation, represented by the rotation matrix $\mathbf{R}_{\text{bone}}$, rotates around the new axis of the bone shown in red in Fig. 3. Thus, the total transformation $\mathbf{R}_{\text{local}}$ is given by:

$$\mathbf{R}_{\text{local}} = \mathbf{R}_{\text{bone}} \mathbf{R}_{\text{joint}}. \tag{2}$$

We refer to this total transformation as $\mathbf{R}_{\text{local}}$ since it denotes the transformation of the joint with respect to its own local coordinate

[1] The limited field of view of each of the cameras and projectors prevents us from capturing the entire height of the human subject.

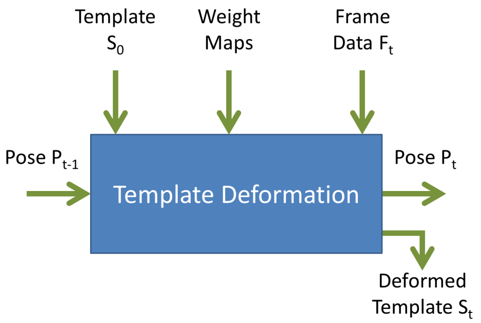


Figure 4: The basic input and output of the iterative mesh deformation method.

frame. Since the rotation $\mathbf{R}_{\text{joint}}$ rotates around an axis that lies in the x-z plane, no component of $\mathbf{R}_{\text{joint}}$ rotates around the y-axis. Therefore, the rotation matrix $\mathbf{R}_{\text{joint}}$ is constrained by the individual minimum and maximum constraints for the x-axis and z-axis. The valid range in which $\mathbf{R}_{\text{joint}}$ can be defined is constrained by the regions that fall within the minimum and maximum angles of rotation for the x-axis, namely $\theta_{\text{min}X}$ and $\theta_{\text{max}X}$, and the z-axis, namely $\theta_{\text{min}Z}$ and $\theta_{\text{max}Z}$. Once the bone has been put in position by $\mathbf{R}_{\text{joint}}$, we constrain the amount of rotation around the bone by using the angle limits for the y-axis, since rotating around the y-axis in the original pose is the same as rotating around the bone axis in the pose after $\mathbf{R}_{\text{joint}}$ is applied. We refer to the minimum and maximum limits of rotation around the bone axis as θ_{minBone} and θ_{maxBone}, respectively.

Even with correct alignment of the skeleton to existing geometry, joint constraints are important. For regions with significant missing geometry due to occlusion, for example, the constraints keep the skeleton in a reasonable pose until the true geometry can be observed again.

Processing a Sequence of Frames

The human subject is captured from only one station at a time at a rate of 60 Hz. At each station, two cameras generate point clouds representing their views of the scene. Each of these two points clouds are meshed into surfaces, which in turn allows for normals to be estimated for each vertex. We merge the six 3D surfaces generated in three consecutive time steps, two from each station, into a partial surface reconstruction of the scene. We refer to the merged partial surfaces captured at time t as a frame F_t.

To capture the overall movement of the human subject over time, the template is sequentially fit to each frame as shown in Fig. 4. The captured data at time t, denoted by F_t, is processed to compute the pose of the human subject at time t, denoted by P_t. We refer to the template in its original pose, as shown in Fig. 1c, as S_0 and the corresponding original skeleton pose itself as P_0. The final skeleton pose that aligns S_0 with frame F_t is referred to as P_t. As seen in Fig. 4, given the template S_0, the captured frame F_t, the weight maps $\mathbf{W}$ and $\bar{\mathbf{W}}$, and the pose P_{t-1} at time $t-1$, we deform the template S_0 to fit the target frame F_t and generate pose P_t and deformed template S_t as output. The steps for template deformation are shown in the flow diagram of Fig. 5. The template deformation takes place in three main steps: initialization, processing, and finalizing. During initialization, all data needed for processing is loaded and the template is deformed to the pose from the previous frame. Within the processing step, the template is evaluated to determine whether it closely fits the target frame. If it does not, the branch fitting loop is executed. During branch fitting, the poses of branches within the skeleton are iteratively updated to align to the target geometry. After each pass through branch fitting, the fit between the source template and the target frame is reevaluated. Once a good fit is found, the processing step is concluded and the finalizing step saves the pose and final template for processing the next frame.

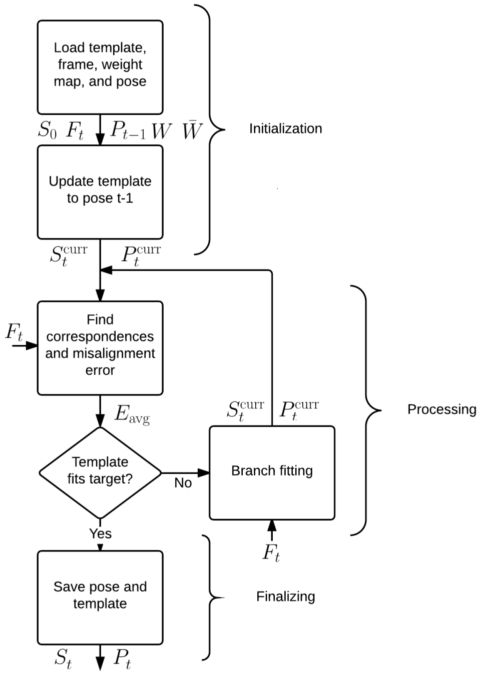

Load template, frame, weight map, and pose
S_0 F_t P_{t-1} W $\bar{W}$
Initialization
Update template to pose t-1
S_t^{curr} P_t^{curr}
F_t
Find correspondences and misalignment error
E_{avg}
S_t^{curr} P_t^{curr}
Processing
Template fits target?
No
Branch fitting
Yes
F_t
Save pose and template
Finalizing
S_t P_t

Figure 5: Processing steps for the template deformation block shown in Fig. 4.

During initialization, the template S_0, vertex weight map $\mathbf{W}$, branch weight map $\bar{\mathbf{W}}$, skeleton pose from the previous time step P_{t-1}, and frame F_t are all loaded. The pose of each bone is represented as a rotation matrix plus translation vector. The pose of a skeleton P_t specifies the transformation of each bone from the original template pose P_0 to its position and orientation at time t, and it is represented by the set of rotation and translation vectors for each bone. While initializing for time instance t, the template S_0 is deformed according to the skeleton pose P_{t-1} that was aligned to frame F_{t-1}. Since the frames are processed sequentially, pose P_{t-1} is the best initial estimate for target frame F_t. The bone weight map $\mathbf{W}$ assigning the influence of each bone on each template vertex is needed to properly deform the template. The vertices of the template are updated to match the skeleton pose using dual quaternion skinning. We refer to this updated template as S_t^{curr} since it is the current best estimate for aligning with F_t.

This deformed template is used as input to the "processing" portion of our algorithm shown in Fig. 5. Correspondences from template S_t^{curr} to the frame F_t are generated, and the average distance between all correspondence points is found. Specifically, a correspondence for each vertex in frame F_t is found by locating the closest vertex in template S_t^{curr} that lies in the direction of the frame vertex's normal. To prevent erroneous correspondences, the maximum distance is kept below a preset threshold. The average misalignment distance over all correspondence pairs is calculated

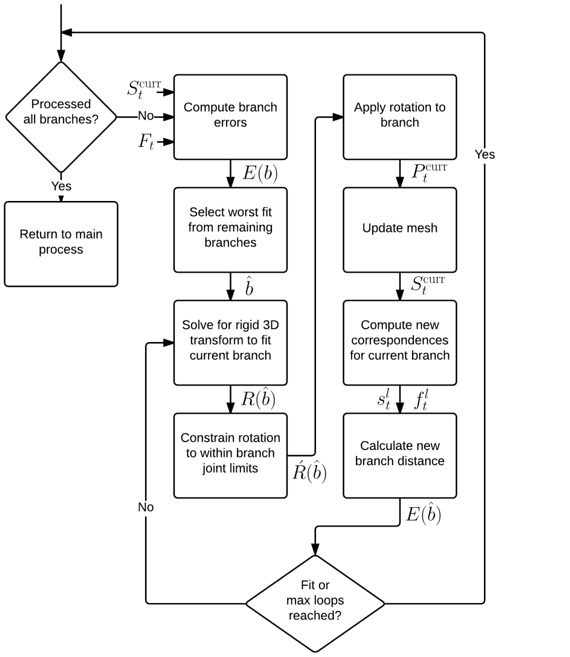


Figure 6: Processing steps for the branch fitting block showed in 5.

and represented as E_{avg}. If this average error is small, then template S_t^{curr} is well aligned with the frame F_t and the finalizing step can start. If S_t^{curr} it not yet well aligned with F_t, then the branch fitting process must be executed until a reasonable fit is found.

In the finalizing step, the pose of the skeleton P_t^{curr} that transforms the source template S_0 into alignment with the frame F_t is stored along with the deformed template S_t^{curr}. The deformed template itself is not used during subsequent processing, but its pose is used during the initialization step of the following frame. The deformed template is the main output of our system corresponding to the dynamic geometry of the human subject.

Branch Fitting

The branch fitting process is illustrated in Fig. 6. At a high level, the pose of each branch in the template skeleton is updated to bring the source template into alignment with the target frame. This process is run until each branch in the skeleton has been updated. The order in which the branches are processed is determined by the misalignment error for each branch, starting with the worst fit branch and continuing in order of decreasing error.

The branch fitting process begins by checking whether all the branches have been processed. If they have not, the error for each branch is calculated. To do this, correspondences between the template S_t^{curr} and captured frame F_t are found as described earlier. We denote the correspondence i as a vertex s_t^i in the template S_t^{curr} and a vertex f_t^i in frame F_t. The Euclidean distances between each corresponding vertex pair in the source and target $d(s_t^i, f_t^i)$ are computed. Using these values, the average misalignment of correspondences of each branch b is

$$E(b) = \frac{\sum\limits_{i \in N} d(s_t^i, f_t^i)\bar{\mathbf{W}}(i,b)}{\sum\limits_{i \in N} \bar{\mathbf{W}}(i,b)}. \quad (3)$$

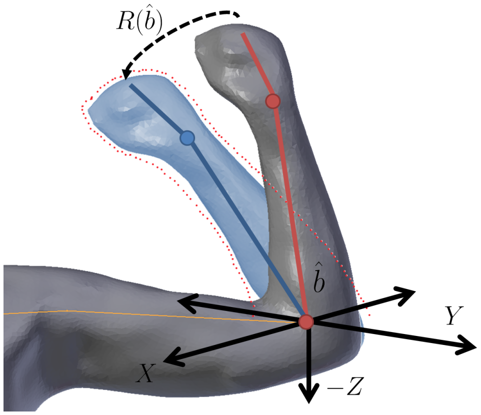


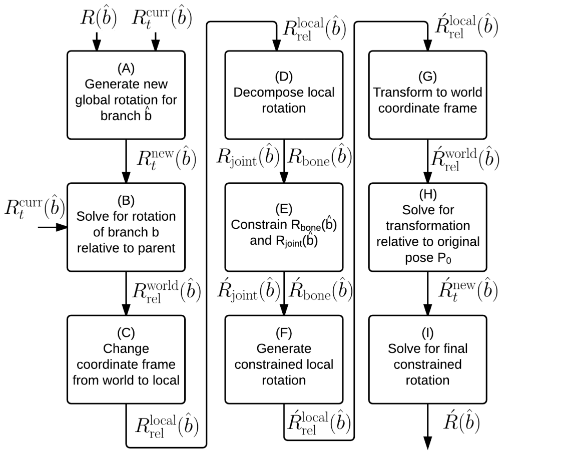
Figure 7: An example of branch fitting. The forearm branch of a template is rotated by $\mathbf{R}(\hat{b})$ to align with points in the target geometry shown as red dots. The new template pose after applying $\mathbf{R}(\hat{b})$ is shown in blue.

Each distance term is weighted by the branch weight map $\bar{\mathbf{W}}(i,b)$, which represents the total influence of branch b on the vertex position s_t^i. We select the most misaligned branch $\hat{b}$ and correct it before any other branch.

The transformation to align the correspondences between the template S_t^{curr} and frame F_t^{curr} in the branch is estimated by solving for a rotation and translation. Specifically, we use a standard 3D rigid transform to align the correspondences for vertices in branch $\hat{b}$. The correspondence weights are proportional to the distance between s_t^l and f_t^l in a correspondence pair. Correspondences with a large distance between the source and target vertices occur in regions where the source template is most misaligned with respect to the target frame, i.e. the regions that are most important to fit. To ensure correct correspondences, we penalize those without similar normals. Specifically, if the normals are more than 90 degrees apart, then they are likely matching the incorrect side of a surface and are thus ignored.

Before computing the rotation to align the correspondences, the vertices of the source and target are translated such that the origin of the world coordinate frame is centered at the lead joint of branch $\hat{b}$. The resulting rotation matrix $\mathbf{R}(\hat{b})$ and translation vector $\vec{T}(\hat{b})$ computed from the 3D transformation represent the update transformation of the branch in the world coordinate system centered at the lead joint of branch $\hat{b}$. Fig. 7 illustrates the forearm of a human template, shown in gray, being fit to target geometry, shown as red dots in the figure. By applying the rotation $\mathbf{R}(\hat{b})$, the forearm of the template is aligned with the target geometry. The rotation $\mathbf{R}(\hat{b})$ is applied in the world coordinate frame centered at the lead joint of branch $\hat{b}$. When fitting all branches other than the root branch, we discard the translation vector $\vec{T}(\hat{b})$ since the position of a branch's lead joint is determined by the parent bone connected to it. For example, in Fig. 1c, the position of the lead joint of bone 7 is determined by the position and orientation of bone 6.

Once the update rotation $\mathbf{R}(\hat{b})$ is estimated, the next step is to ensure that the rotation is within the constrained limit for the branch. Constraining the update rotation of a branch results in a new rotation matrix $\acute{\mathbf{R}}(\hat{b})$. With the new rotation, the position of branch $\hat{b}$ is updated in P_t^{curr} and the template S_t^{curr} is updated according to this new pose. The template S_t^{curr} is deformed using



$R(\hat{b})$
$R_t^{\text{curr}}(\hat{b})$
(A) Generate new global rotation for branch $\hat{b}$
$R_t^{\text{new}}(\hat{b})$
$R_t^{\text{curr}}(\hat{b})$
(B) Solve for rotation of branch b relative to parent
$R_{\text{rel}}^{\text{world}}(\hat{b})$
(C) Change coordinate frame from world to local
$R_{\text{rel}}^{\text{local}}(\hat{b})$
$R_{\text{rel}}^{\text{local}}(\hat{b})$
(D) Decompose local rotation
$R_{\text{joint}}(\hat{b})$
$R_{\text{bone}}(\hat{b})$
(E) Constrain $R_{\text{bone}}(\hat{b})$ and $R_{\text{joint}}(\hat{b})$
$\acute{R}_{\text{joint}}(\hat{b})$
$\acute{R}_{\text{bone}}(\hat{b})$
(F) Generate constrained local rotation
$\acute{R}_{\text{rel}}^{\text{local}}(\hat{b})$
$\acute{R}_{\text{rel}}^{\text{local}}(\hat{b})$
(G) Transform to world coordinate frame
$\acute{R}_{\text{rel}}^{\text{world}}(\hat{b})$
(H) Solve for transformation relative to original pose P_0
$\acute{R}_t^{\text{new}}(\hat{b})$
(I) Solve for final constrained rotation
$\acute{R}(\hat{b})$

Figure 8: Processing steps for the constraining a branch rotation block shown in 6.

dual quaternion skinning.

Given the new pose for the template S_t^{curr}, we recompute the correspondences for the updated branch $\hat{b}$, the one just rotated, and compute the average misalignment error for the branch using Equation 3. The process is repeated until the average misalignment error for $\hat{b}$ drops below a certain threshold for a preset number of iterations. Once this happens, we return to the beginning of the branch fitting process and check to see whether all branches have been fit. After all branches are processed, we exit the branch fitting process.

Results

We test our deformation method on a 50-second animation sequence consisting of 992 frames. Starting from the grayscale images that capture the human subject under structured light, a sequence of phase images are calculated for each camera. These phase images are unwrapped as in [7] and a sequence of point clouds are generated from each camera. The point clouds from each of the six geometry capturing cameras are filtered, meshed, and transformed into a single world coordinate frame. The result is a sequence of partial scans that we use to deform our template. During processing, we set parameters to iterate through all branches four times per frame. Additionally, the rigid transformation of each branch is estimated at most four times before proceeding to the other branches. Fig. 9 shows the transformation of each branch over time in the local coordinate frame. The branches are labeled as in Fig. 1c and the rotation angles θ_X, θ_Z, and θ_{bone} of each branch are relative to the branch's orientation in pose P_0. Specifically, the angles for a bone b are derived from the rotation matrices $\acute{\mathbf{R}}_{\text{joint}}(b)$ and $\acute{\mathbf{R}}_{\text{bone}}(b)$ computed from the final pose P_t of each frame shown in Fig. 4. The sequence of captured geometry starts with the human subject in pose P_0, which is reflected in Fig. 9 as the θ_X, θ_Z, and θ_{bone} angles for each branch set to zero. Over time, the angles drift from this pose. The angles for the wrists, i.e. branches 7 and 11, are not illustrated because we did not allow for rotation around these joints in the final reconstruction. Limited geometry captured for the hands in most frames makes it difficult to constrain their movement consistently. As seen, there is a significant variation in the amount of rotation for each branch. The branches in the arms, specifically branches 5, 6, 9, and 10, exhibit the greatest variation in orientation.

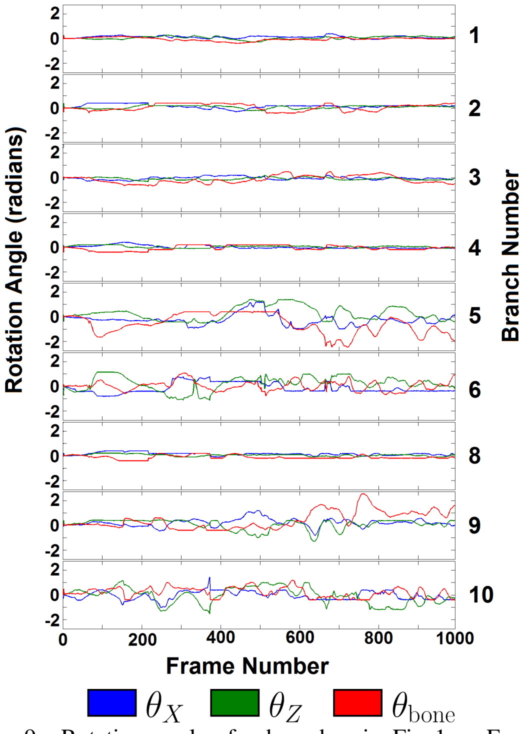


Figure 9: Rotation angles for branches in Fig. 1c. For each branch, the sequence of values for θ_X, θ_Z, and θ_{bone} are plotted over the frame indices.

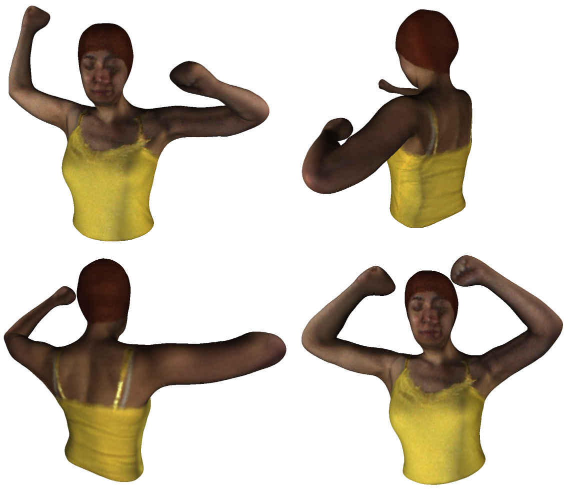


Figure 10: The template is deformed to fit the partial scans in a variety of poses.

As shown in Fig. 10, our deformation method is able to successfully capture the human subject in a variety of poses. The human subject is able to rotate and change the direction her body faces while still having the template successfully deform to match the geometry. The deformation processing of each frame takes less than two minutes with non-optimized prototype code. We expect this to be significantly improved by using the GPU.

The rendered view of the reconstructed sequence is shown in supplemental videos A and B. Specifically, video A shows a side-by-side comparison of a video of the human subject and the same sequence reconstructed using our proposed approach. Also, video B shows a 360-degree virtual view of the subject. As seen, our method is able to both accurately capture the geometry of the human subject. In video A, we can clearly observe our reconstructed geometry matching the motion of the human subject. In video B, the video shows that the motion is accurately captured on all sides of the human subject.

Conclusion

We have presented a multi-view structured-light system capable of performing markerless motion capture for human subjects. With rich captured 3D data rather than 2D video streams, we are able to easily deform a template to match partial scans of the human subject. Unlike existing methods, we can easily create templates directly from our system without having to rely on a special scanning process or needing high quality laser scans.

In future work, we plan to improve the quality and level of details of geometry resulting from the system. This could be done by testing other configurations of structured-light patterns or changing projector and camera hardware for models with higher speed and improved resolutions. This would allow us to also integrate local deformation methods to capture the subtle details of the human subject's geometry [12]. Additionally, we would like to decrease processing time by offloading some computing onto the GPU and by streamlining the 3D geometry generation and deformation steps.

References

[1] D. Anguelov, P. Srinivasan, D. Koller, S. Thrun, J. Rodgers, and J. Davis. Scape: Shape completion and animation of people. *ACM Trans. Graph.*, 24(3):408–416, July 2005.

[2] A. Baak, M. Mller, G. Bharaj, H.-P. Seidel, and C. Theobalt. A data-driven approach for real-time full body pose reconstruction from a depth camera. Advances in Computer Vision and Pattern Recognition, pages 71–98. Springer London, 2013.

[3] S. Corazza, L. Mndermann, E. Gambaretto, G. Ferrigno, and T. Andriacchi. Markerless motion capture through visual hull, articulated icp and subject specific model generation. *International Journal of Computer Vision*, 87(1-2):156–169, 2010.

[4] E. de Aguiar, C. Stoll, C. Theobalt, N. Ahmed, H.-P. Seidel, and S. Thrun. Performance capture from sparse multi-view video. *ACM Trans. Graph.*, 27(3):98:1–98:10, 8 2008.

[5] M. Dou, H. Fuchs, and J.-M. Frahm. Scanning and tracking dynamic objects with commodity depth cameras. In *Mixed and Augmented Reality (ISMAR), 2013 IEEE International Symposium on*, pages 99–106, 10 2013.

[6] J. Gall, C. Stoll, E. de Aguiar, C. Theobalt, B. Rosenhahn, and H.-P. Seidel. Motion capture using joint skeleton tracking and surface estimation. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 1746–1753, 6 2009.

[7] R. R. Garcia and A. Zakhor. Consistent stereo-assisted absolute phase unwrapping methods for structured light systems. *Selected Topics in Signal Processing, IEEE Journal of*, 6(5):411–424, 9 2012.

[8] R. R. Garcia and A. Zakhor. Geometric calibration for a multi-camera-projector system. In *Applications of Computer Vision (WACV), 2013 IEEE Workshop on*, pages 467–474, 1 2013.

[9] A. Griesser, T. Koninckx, and L. Van Gool. Adaptive real-time 3D acquisition and contour tracking within a multiple structured light system. In *Computer Graphics and Applications. Proceedings. 12th Pacific Conference on*, pages 361–370, 10 2004.

[10] L. Kavan, , S. Collins, J. Žára, and C. O'Sullivan. Skinning with dual quaternions. In *Proceedings of the 2007 Symposium on Interactive 3D Graphics and Games*, I3D '07, pages 39–46. ACM, 2007.

[11] M. Kazhdan, M. Bolitho, and H. Hoppe. Poisson surface reconstruction. In *Proceedings of the Fourth Eurographics Symposium on Geometry processing*, 2006.

[12] H. Li, B. Adams, L. J. Guibas, and M. Pauly. Robust single-view geometry and motion reconstruction. *ACM Transactions on Graphics (Proceedings SIGGRAPH Asia 2009)*, 28(5), December 2009.

[13] H. Li, L. Luo, D. Vlasic, P. Peers, J. Popović, M. Pauly, and S. Rusinkiewicz. Temporally coherent completion of dynamic shapes. *ACM Trans. Graph.*, 31(1):2:1–2:11, 2 2012.

[14] M. Liao, Q. Zhang, H. Wang, R. Yang, and M. Gong. Modeling deformable objects from a single depth camera. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 167–174, Sept 2009.

[15] N. J. Mitra, S. Flöry, M. Ovsjanikov, N. Gelfand, L. Guibas, and H. Pottmann. Dynamic geometry registration. In *Proceedings of the Fifth Eurographics Symposium on Geometry Processing*, SGP '07, pages 173–182, 2007.

[16] T. B. Moeslund and E. Granum. A survey of computer vision-based human motion capture. *Computer Vision and Image Understanding*, 81(3):231–268, 2001.

[17] T. B. Moeslund, A. Hilton, and V. Krger. A survey of advances in vision-based human motion capture and analysis. *Computer Vision and Image Understanding*, 104(23):90–126, 2006.

[18] T. Popa, I. South-Dickinson, D. Bradley, A. Sheffer, and W. Heidrich. Globally consistent space-time reconstruction. *Computer Graphics Forum*, 29(5):1633–1642, 2010.

[19] J. Starck and A. Hilton. Surface capture for performance-based animation. *Computer Graphics and Applications, IEEE*, 27(3):21–31, May 2007.

[20] R. W. Sumner, J. Schmid, and M. Pauly. Embedded deformation for shape manipulation. In *ACM SIGGRAPH 2007 Papers*, SIGGRAPH '07. ACM, 2007.

[21] J. Süßmuth, M. Winter, and G. Greiner. Reconstructing animated meshes from time-varying point clouds. *Computer Graphics Forum*, 27(5):1469–1476, 2008.

[22] C. Theobalt, N. Ahmed, H. Lensch, M. Magnor, and H.-P. Seidel. Seeing people in different light-joint shape, motion, and reflectance capture. *Visualization and Computer Graphics, IEEE Transactions on*, 13(4):663–674, 7 2007.

[23] D. Vlasic, I. Baran, W. Matusik, and J. Popović. Articulated mesh animation from multi-view silhouettes. In *ACM SIGGRAPH 2008 Papers*, SIGGRAPH '08, pages 97:1–97:9. ACM, 2008.

[24] D. Vlasic, P. Peers, I. Baran, P. Debevec, J. Popović, S. Rusinkiewicz, and W. Matusik. Dynamic shape capture using multi-view photometric stereo. In *ACM SIGGRAPH Asia 2009 Papers*, SIGGRAPH Asia '09, pages 174:1–174:11. ACM, 2009.

[25] T. Weise, B. Leibe, and L. Van Gool. Fast 3d scanning with automatic motion compensation. In *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, pages 1–8, 6 2007.

[26] Q. Zhang, B. Fu, M. Ye, and R. Yang. Quality dynamic human body modeling using a single low-cost depth camera. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 676–683, June 2014.

[27] S. Zhang and P. Huang. High-resolution, real-time 3d shape acquisition. In *Computer Vision and Pattern Recognition Workshop, 2004. CVPRW '04. Conference on*, pages 28–28, 6 2004.