POSE-NORMALIZED VIEW SYNTHESIS FROM SILHOUETTES

Zhanfeng Yue and Rama Chellappa

Center for Automation Research
University of Maryland
College Park, Maryland, 20742
{zyue,rama}@cfar.umd.edu

ABSTRACT

In this paper, we introduce an active view synthesis approach from silhouettes. With the virtual camera moving on a properly selected circular trajectory around an object of interest, we achieve a collection of virtual views of the object, which is equivalent to the case that the object is on a rotating turntable and captured by a static camera whose optical axis is parallel to the turntable. We show how to derive the virtual camera's extrinsic parameters at each position on the trajectory. Using the turning function distance as the silhouette similarity measurement, this approach can be used to generate the desired pose-normalized images for recognition applications.

1. INTRODUCTION

Recently, image-based rendering (IBR) has become an emerging and competing rendering paradigm. Given an observing direction, the IBR technique is able to synthesize the corresponding view of an object of interest without recovering its 3D structure. With an active virtual camera, images from different viewpoints can be generated to give us a good understanding of the object. In this paper, we show how to generate a collection of the object's images (turntable image collection) which are captured by a camera moving around the object, with the optical axis parallel to the plane that the object stands on. Using a small number of widely-placed views as input, the turntable image collection can be rendered fast and efficiently using the image based visual hull technique. Upon this synthesized image collection, we are able to produce a pose-normalized video sequence by comparing the turning function distance.

In contrast to the traditional geometry-based rendering, IBR techniques rely on view interpolation, or pixel re-projection from source images onto the target image, with the rendering speed independent of scene complexity [7]. It is usually required that views must be close enough so that correspondences across these views are easy to establish. Also correspondences must be maintained over many views which span large changes. An alternative approach is based on constructing the volumes or surfaces in 3D space that are consistent with input images [4]. These methods generally depend on calibrated cameras.

A *Visual Hull* (VH) of an object is the intersection of all the extruded cone-like shapes that result from lifting the silhouettes in all views [8]. It is possible to reduce the computation of VH to 2D operations since it contains only points that project onto the silhouettes. *Image Based Visual Hull* (IBVH) [9] is an efficient geometrically-valid pixel reprojection method to compute VH, with no need to find the wide baseline correspondence. The algorithm is able to render a desired view of n^2 pixels in $O(kn^2)$ where k is the number of input images. Fig. 1 shows an example of view synthesis with IBVH.

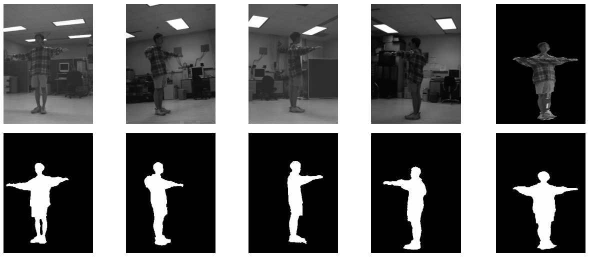
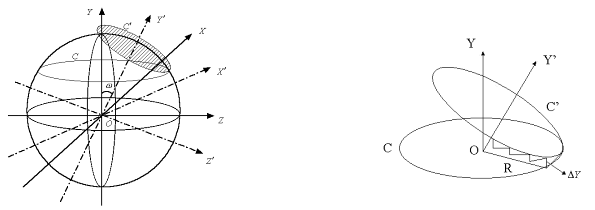


Fig. 1. (a) An example of IBVH: the images observed from the 4 static cameras (top) and corresponding silhouette images (bottom). (b) The rendered image corresponding to a novel view with texture (top) and without texture (bottom) obtained with IBVH.

One important application for image rendering is in human identification under variations. The varying appearances due to different poses can make the recognition problem very challenging. Using image rendering techniques to synthesize pose-invariant images is an appealing approach. Some promising results have been reported for integrated gait and face recognition from multiple views [11]. A strong assumption made in [11] is that the person is walking and generally facing forward. Under this assumption, the person's motion trajectory is easy to estimate and the virtual camera can be placed accordingly. This approach will not work if the motion trajectory is hard to estimate, or not available (e.g., turning around). In order to attack the general motion case, we resort to viewpoint selection technique which is rapidly becoming a key issue in computer graphics.

The view sphere of an object is a sphere which is centered at the object and has a fixed radius [2]. We propose to use view sphere and the silhouette turning function distance to generate pose-normalized views for recognition applications. By moving the virtual camera along a properly selected circular trajectory on the view sphere, the turntable image collection can be rendered quickly and efficiently with IBVH. We derive a method to align the camera calibration coordinate system and the world coordinate system if they do not coincide each other, so that the virtual camera's position on the trajectory can be decided accurately. Considering the nonrigid free-form human movement, the silhouette turning function distance is used to select the desired viewpoint and generate the pose-normalized views.



(a) (b)

Fig. 2. (a) The calibration coordinate system does not coincide with the world coordinate system, so the virtual camera's circular trajectory should be the shaded C' which is not parallel to the X-Z plane. (b) With the silhouette centroid observed for the previous position, the circle C' can be approximated by adjusting the Y coordinate of the virtual camera.

The remainder of this paper is organized as follows. In section 2 we present the active virtual camera positioning scheme. Section 3 describes how to produce the pose-normalized views using the rendered turntable image collection and the pre-built silhouette turning functions for known viewing directions. Conclusions are presented in section 4.

2. TURNTABLE IMAGE COLLECTION RENDERING WITH IMAGE BASED VISUAL HULL

Assuming the virtual camera's intrinsic parameters are same as the given real camera, we need to know its extrinsic parameters in order to decide its position along the designated trajectory. There are two coordinate systems to consider: the calibration coordinate system is the one with which the camera calibration information is recorded for the input views, and the world coordinate system is the one with the Y-axis perpendicular to the ground plane.

The camera calibration coordinate system is used for the virtual camera, except that the origin is set as the 3D centroid O of the computed VH. Apparently, this is not a static coordinate system because the origin changes with the centroid of the VH from frame to frame. The extrinsic parameters to be determined include the translation vector $[T_x, T_y, T_z]^T$ and the rotation angles $[\phi, \beta, \psi]$ (pitch, yaw and roll, respectively) around the X, Y, and Z-axis respectively.

If the calibration coordinate system coincides with the world coordinate system, then the virtual camera's circular motion trajectory is parallel to the X-Z plane. Starting from an initial position $P(X_p, Y_p, Z_p)$, and with the view sphere setting up around O with the radius $R = \|\vec{P} - \vec{O}\|$, the virtual camera's circular trajectory is centered at $(0, Y_p, 0)$, and has the radius $r = \sqrt{R^2 - Y_p^2}$, with Y_p, ϕ and ψ not changing along the circle. Given β, X and Z coordinates can be determined uniquely if r is fixed. Hence, θ is the only parameter we need to control. Let $\Delta\beta$ be the step size for β, X_n and Z_n denote the X and Z coordinates for position n. It is easy to derive

$$X_{n+1} = X_n + 2r\text{sin}(\Delta\beta/2)\text{sin}\,\alpha \tag{1}$$

$$Z_{n+1} = Z_n - 2r\text{sin}(\Delta\beta/2)\text{cos}\,\alpha \tag{2}$$

where $\alpha = |\beta| - \Delta\beta/2$ is an auxiliary angle.

When accurate calibration tools, such as the Peak Performance calibration frame [3], are used, it often happens that the calibration coordinate system does not coincide with the world coordinate system, but has an unknown angle ω between its Y-axis and the world coordinate system's Y-axis, as shown in Fig. 2(a). In this case, if the virtual camera still moves along the circle parallel to the X-Z plane, we can observe that the object in the collected images keeps moving upward in the first half circle and downward in the second half. Also the x-coordinate of the object's 2D centroid does not remain fixed. To solve this problem, we need to align the calibration coordinate system with the world coordinate system. Assuming that the object stands upright on the ground plane, ω can be estimated by solving an optimization problem, with the object's vertical principal axes in each input image as corresponding 2D lines. Here we propose another feasible solution without estimating ω. Let $p_n = (x_n, y_n)$ be the object's 2D centroid observed at position n, and Δy_n and Δx_n the change in y_n and x_n from position n to position $n+1$ respectively. So we have $\Delta y_n = y_{n+1} - y_n$ and $\Delta x_n = x_{n+1} - x_n$. We try to approximate the designated circle C' with Δy_n and Δx_n observed, as shown in Fig. 2(b). Let ΔY_n be the virtual camera's displacement along the Y direction from position n to position $n+1$. Using the property of similar triangle we have

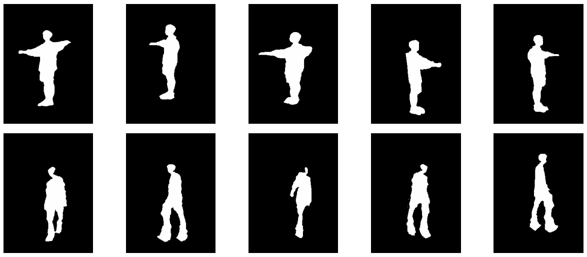


Fig. 3. The synthesized turntable silhouette image collection. Top: turning and pointing sequence taken at Keck lab. Bottom: walking sequence collected at MIT AI lab.

$$\frac{\Delta y_n dp_y}{\Delta Y_n} = \frac{f}{R}, \frac{\Delta x_n dp_x}{\Delta D_n} = \frac{f}{R} \tag{3}$$

where dp_y is the y-size per pixel, dp_x is the x-size per pixel, f is the camera's focal length, and ΔD_n is the translation adjustment on the circle C in order to keep $x_{n+1} = x_n$. ΔD_n can be compensated by adjusting β accordingly. Since ΔD_n is very small compared to the sphere radius R, the adjusting angle $\Delta\theta$ can be approximated as $\Delta\theta = 2\arctan\dfrac{\Delta D_n}{2R}$.

In order to keep the view sphere radius R constant, X and Z coordinates have to be further adjusted based on ΔY_n. Denote $r' = \sqrt{R^2 - (Y_n + \Delta Y_n)^2}$ and $\Delta r = |r - r'|$, then $\Delta X_n = \Delta r \cos|\beta|$ and $\Delta Z_n = \Delta r \sin|\beta|$. So in addition to (1) and (2), we have

$$\begin{cases} X_{n+1} = X_n - \Delta X,\ Z_{n+1} = Z_n + \Delta Z, & \text{if } Y_n \cdot \Delta Y_n \geq 0 \\ X_{n+1} = X_n + \Delta X,\ Z_{n+1} = Z_n - \Delta Z, & \text{otherwise} \end{cases} \tag{4}$$

The roll angle ψ also has to be modified to keep the object's principal axes perpendicular and parallel to the ground plane. At position n, the eigenvector $[\vec{e_x}, \vec{e_y}]$ of the silhouette image is computed and we have $\psi_{n+1} = \psi_n + \arctan e_{x1}/e_{x2}$.

The active virtual camera positioning algorithm can be summarized as follows:

1. Choose the extrinsic parameters of the virtual camera as the average of any two real cameras' parameters. Usually this is a good initial position. Select step size $\Delta\beta$ for β.

2. Get the silhouette image at the current position n, and compute the 2D centroid changes Δx_n and Δy_n from the silhouette image at the previous position. Compute the eigenvector $[\vec{e_x}, \vec{e_y}]$ of the current silhouette image. Compute ΔY_n and ΔD_n as in (3), then compute $\Delta\theta$ with ΔD_n. Let $\psi_{n+1} = \psi_n + \arctan(e_{x1}/e_{x2})$. Modify X and Z coordinates using (4) with ΔY_n obtained in the previous step.
3. Move the virtual camera to the next position using (1) and (2), and let $\beta_{n+1} = \beta_n + \Delta\beta$.
4. Repeat steps 2 through 3 until the virtual camera comes back to the original position.

This algorithm was implemented and tested with several sequences. The input is the synchronized perspective 4-view silhouette sequences of a person, with the cameras fully calibrated. The output is the rendered turntable image collection of the person for each frame. The turning and pointing sequence was taken at the Keck lab in University of Maryland. The person's motion is mainly turning motion, so the trajectory information is hard to extract from the sequence. The top row of Fig. 3 is the result for the pointing and turning sequence. The normal walking sequence was collected at MIT AI lab. The trajectory information can be estimated with VH 3D centroid coordinates as mentioned in [11]. Our algorithm also works well as shown in the bottom row of Fig. 3. In both sequences, $\Delta\beta = 0.3rad$, so there are 21 positions on the whole circle around the person.

3. DESIRED VIEW SELECTION USING SILHOUETTE TURNING FUNCTION DISTANCE

In order to select the desired view from the turntable image collection, we need to compare the turntable images with the knowledge base of silhouettes associated with known viewing directions. In [5] a template matching method is proposed to estimate the human pose from silhouettes, where a body posture is represented by the normalized horizontal and vertical projection histograms, the median coordinate, and the major axis of its silhouette. The resulting silhouette is compared with the projection templates using the sum of absolute difference method to estimate the main posture. While this method is simple and fast, it is not robust and produces some ambiguities. Therefore a distance which can measure the similarity of two silhouettes is needed. According to [1], this distance should satisfy a number of properties, including 1) it should be a metric, 2) it should be invariant under translation, rotation and change-of-scale, 3) it should be reasonably easy to compute, and 4) it should match our intuition. To compare a shape A, which is stored as a model (in our case, the knowledge base of silhouettes associated with known viewing directions), with a shape B, which is found to exist in an image (in our case, the turntable images), the distance between the turning functions $\Theta_A(s)$ and $\Theta_B(s)$ is an efficient measurement of the similarity.

The turning function $\Theta_A(s)$ measures the angle of the counterclockwise tangent as a function of the arc-length s measured from some reference point O on A's boundary. $\Theta_A(s)$ keeps track of the turning that takes place, increasing with left-hand turns and decreasing with right-hand turns. The turning function measures the turning that takes place as you move along the perimeter. Mathematically, if $\kappa(s)$ is the curvature function of a curve then $\Theta(s) = \int \kappa(s)$.

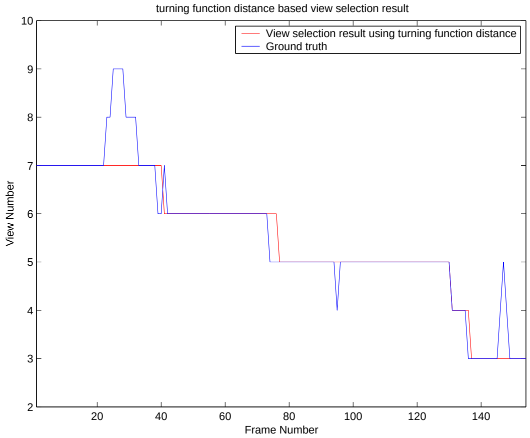


Fig. 4. The view selection result comparison for the turning and pointing sequence, with the blue curve indicating the view selection result using the turning function distance, and the red curve indicating the ground truth.

In [1], the distance function between two polygons A and B is formally defined as the L_p distance between their two turning functions $\Theta_A(s)$ and $\Theta_B(s)$, minimized with respect to rotation θ and choice of reference points t,

$$d_p(A,B) = (\min_{\theta\in\Re, t\in[0,1]} \int_0^1 |\Theta_A(s+t) - \Theta_B(s) + \theta|^p ds)^{\frac{1}{p}}$$

$$= (\min_{\theta\in\Re, t\in[0,1]} D_p^{A,B}(t,\theta))^{\frac{1}{p}},$$

where $D_p^{A,B}(t,\theta) = \int_0^1 |\Theta_A(s+t) - \Theta_B(s) + \theta|^p ds$. If the L_2 metric is used, the authors proved that the distance $d_2(A,B)$ between two polygons A and B (with m and n vertices) can be computed exactly in time $O(mn \log mn)$.

The turning function metric has been shown to correlate well with human notions of shape similarity [10]. Howe [6] used both the turning function and the Chamfer distance for silhouette lookup for automatic pose tracking. In implementing the turning function distance, we use the method mentioned in [10], where dynamic programming is used to account for warpings that may exist between the query object and database object that result in stretching and compression. It is quite possible that the matching between the points along the border of shape A and the points along the border of shape B is not one-to-one, but one-to-many or many-to-one. It computes the global best match between $\Theta_A(s)$ and $\Theta_B(s)$ in the sense that it pairs up each element of $\Theta_A(s)$ with an element of $\Theta_B(s)$(and vice versa), but the matching must proceed monotonically through both sets. Thus it computes two sequences $i_1, i_2, \ldots, i_k$ and $j_1, j_2, \ldots, j_k$ such that either $i_{t+1} = i_t$ or $i_{t+1} = i_t + 1$ (similarly for j), by normalizing the distance between matched turning angle points:

$$D = \sum_{t=1,2,\ldots,k} |i_t - j_t|$$

The knowledge base of silhouettes consists of the turning functions $\Theta_{A_i}(s)$ of the silhouettes for some canonical poses, for example, the five standard stances in a human walk cycle. By definition, the turning function is invariant under translation and scaling of the polygon A_i. Therefore normalization is not necessary in building the knowledge base. The turning function $\Theta_{B_j}(s)$ of the silhouette at current viewing direction is calculated and the distance functions $d_2(A_i, B_j)$ between $\Theta_{A_i}(s)$ and $\Theta_{B_j}(s)$ are

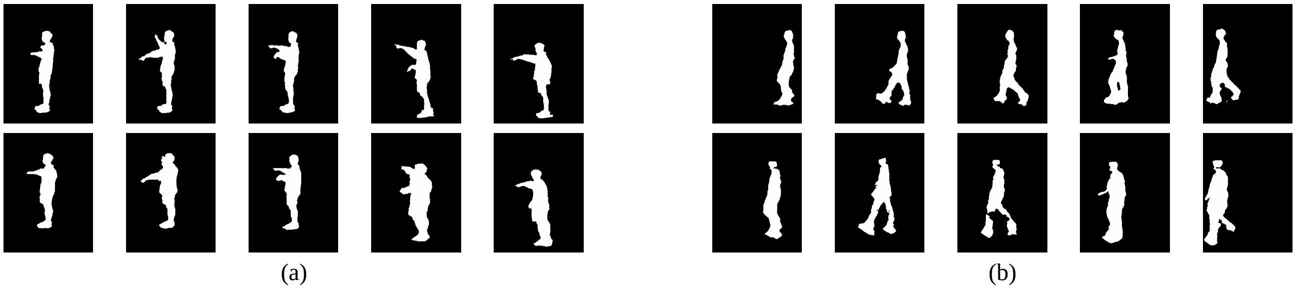


Fig. 5. (a) The side view ground truth for the Keck lab sequence (top) and the rendered side view for the Keck lab sequence (bottom). (b) The side view ground truth for the MIT sequence (top) and the rendered side view for the MIT sequence (bottom).

computed. In addition to the silhouette B_j at the current viewing direction, we can get an auxiliary silhouette C_j by placing the virtual camera at the position where the angle around the Y-axis has $\pi/2$ difference with the current position. Take the example that the desired view is side view, let $d_2(SA_i, B_j)$ be the turning function distance between B_j and the standard stances SA_i for the side view, and $d_2(FA_i, C_j)$ be the distance between C_j and the standard stances FA_i for the frontal view, then the final decision measurement is $S(i, j) = d_2(SA_i, B_j) + d_2(FA_i, C_j)$. The view with the minimal distance not only gives the side view, but also gives the stance at which the person stands for the current frame. With this method, the desired view is selected only when it has a small distance in matching the side view stance and its auxiliary silhouette has a small distance in matching the frontal view stance at the same time. This greatly reduces the possible ambiguities when only the side view itself is considered.

Usually there will be no abrupt change from one frame to the next, so we do not need to generate all the virtual views around the person for each frame. Only a small number of neighboring positions of the selected view in the previous frame is synthesized and compared. Experiments show that the results are good enough while the speed is much faster compared to generate all the virtual views around the person for each frame. As we can see from Fig. 4, which shows the view selection result for the turning and pointing sequence using the turning function distance, the selected view follows the ground truth quite well for most of the frames. Although some error still exists for several frames, it disappears in the next 2-3 frames. Fig. 5 shows the virtual side views for the turning and pointing sequence and the normal walking sequence.

4. CONCLUSIONS

We have described a pose-normalized view synthesis approach from silhouettes. A collection of virtual images of the object can be obtained by properly moving the virtual camera along a circular trajectory. We showed how to derive the virtual camera's translation and rotation at each position on the trajectory. Silhouette turning function distance is combined with active image rendering approach to get the pose-normalized views for recognition applications.

ACKNOWLEDGEMENTS

We are grateful to Dr. Nicholas R. Howe for providing the code capturing the turning functions.

5. REFERENCES

[1] E. Arkin, L. Chew, D. Huttenlocher, K. Kedem, and J. Mitchell, “An Efficiently Computable Metric for Comparing Polygonal Shapes,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(3):209–216, 1991.

[2] P. Barral, G. Dorme, and D. Plemenos, “Visual Understanding of a Scene by Automatic Movement of a Camera,” *International Conference GraphiCon’99*, 1999.

[3] L. Davis, E. Borovikov, R. Cutler, D. Harwood and T. Horprasert, “Multi-perspective Analysis of Human Action” *Proc. Third International Workshop on Cooperative Distributed Vision*, November, 1999

[4] C. Dyer, “Volumetric Scene Reconstruction from Multiple Views,” *Foundations of Image Understanding*, The Kluwer International Series in Engineering and Computer Science, ISBN 0-7923-7457-6, pages 469-489, 2001.

[5] I. Haritaoglu, D. Harwood, and L. S. Davis, “Ghost: A Human Body Part Labeling System Using Silhouettes,” *Proc. Int. Conf. on Pattern Recognition*, pages 77–82, 1998.

[6] N. Howe, “Silhouette Lookup for Automatic Pose Tracking,” *IEEE Workshop on Articulated and Nonrigid Motion*, June, 2004.

[7] S. Kang, “A Survey of Image-Based Rendering Techniques,” *SPIE*, 3641:2–16, January, 1999.

[8] A. Laurentini, “The Visual Hull Concept for Silhouette-based Image Understanding,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(2):150–162, 1994.

[9] W. Matusik, C. Buehler, R. Raskar, S. J. Gortler, and L. McMillan, “Image-Based Visual Hulls,” *Proc. SIGGRAPH 2000*, pages 369–374, 2000.

[10] B. Scassellati, S. Alexopoulos, and M. Flickner, “Retrieving Images by 2D shape: A Comparison of Computation Methods with Human Perceptual Judements ,” *Storage and Retrieval for Image and Video Databases* , pages 2–14, 1994.

[11] G. Shakhnarovich, L. Lee, and T. Darrell, “Integrated Face and Gait Recognition from Multiple Views,” *Proc. Computer Vision and Pattern Recognition 2001*, pages 439–446, 2001.