International Conference on Communication, Media, Technology and Design

ICCMTD
09-11 May 2012
Istanbul - Turkey

VISUALIZING TIME COHERENT THREE-DIMENSIONAL CONTENT USING ONE OR MORE MICROSOFT KINECT CAMERAS

Naveed Ahmed
University of Sharjah
Sharjah, United Arab Emirates

Abstract
Visualizing or digitization of the real world is one of the major goals of Computer Graphics. To facilitate this visualization, a number of techniques have been proposed that capture the shape, appearance and many other attributes of real world objects. This visualization leads to a number of benefits in a number of areas, e.g. interactive media, digital art, multimedia systems, education, mass communications etc. If one needs to visualize dynamic real-world objects, it is required to capture not only the shape and appearance but also the motion of the objects. This is normally achieved using video cameras but the captured attributes are not coherent over time as each frame of the video is independent of the other. In this paper, we present a new method for capturing coherent features of a three-dimensional representation of a real-world dynamic object that is recorded from standard one or more color or Microsoft Kinect cameras. We show that using both color and depth information of a real world object, it is not only possible to capture the dynamic real world objects but it is also possible to visualize the dynamic data in a time-coherent manner by acquiring the coherence information directly from the data. Our work can be employed in a number of scenarios to enhance the visual representation of a dynamic real-world scene.
Keywords: Multi-view video, dynamic scene reconstruction, three-dimensional scene geometry*.*

INTRODUCTION

We present a new method to estimate time-coherent information from a moving three-dimensional point cloud of a real world object. This three-dimensional content can be obtained by either directly through one or more Microsoft Kinect cameras or using a multi-view color acquisition system. We show that both image and depth information obtained from one or more Microsoft Kinect cameras or a multi-view color acquisition system can be utilized to construct a temporal structure between time-varying dynamic scene geometry. A three-dimensional representation of a dynamic object is used in a number of applications, e.g. motion capture, free-viewpoint video, dynamic scene reconstruction, scene analysis etc. Traditionally the multi-view video recordings are acquired using synchronized color cameras that are placed around a moving object. These multi-view recordings were then used to reconstruct three-dimensional information of the dynamic real-world scene.

One of the earlier works in this area was presented by Carranza et al. [6], who used eight multi-view recordings to reconstruct the motion and shape of a moving subject and applied it in the area of free-viewpoint video reconstruction. In the following years this work was extended by Theobalt et al. [15] so that in addition to capturing the shape and motion they also captured surface reflectance properties of a dynamic object. Starck et al. [14] presented a high quality surface reconstruction method that could capture detailed moving geometry from multi-view video recordings. Later de Aguiar et al. [7] and Vlasic et al. [16] presented new method for reconstructing really high quality of dynamic scene using multi-view video recordings. Both of their methods first obtained the shape of the real world object using a laser scanner and then deformed the shape to reconstruct the 3D animation. Ahmed et al. [1] presented a method of dynamic scene reconstruction with time coherent information without the use of any template geometry, but unlike our method they did not explicitly include multiple matching criteria for extracting time coherence in their method.

Recently, acquiring dynamic three-dimensional content is even easier with the arrival of low cost depth cameras, esp. Microsoft Kinect [13]. A number of new methods are proposed that make use of these depth sensors to reconstruct three-dimensional scene geometry. One of the main benefit of using Microsoft Kinect is that it gives both color and depth information of the scene, whereas with only color cameras the depth information has to be obtained by means of finding some correspondence between the cameras. One of the applications of this is the reconstruction of visual hulls which was exploited by [1]. The depth information can also be obtained by means of other type of sensors, e.g. Time of Flight sensors [11].

One or more depth sensors are used for the reconstruction of both static and dynamic objects. Kim et al. [10] and Castaneda et al. [5] presented method of reconstructing a three-dimensional representation of a static object using depth cameras. Berger et al. [3], Girshich et al. [8], Weiss et al. [17], and Baak et al. [2] used depth cameras for reconstructing three-dimensional shape, pose and motion. They demonstrate that it is possible to get good quality results by employing both depth and color cameras.

For capturing the dynamic scene data using depth sensors, two methods were recently presented by Kim et al. [11] and Berger et al. [3]. The former work employs RGB cameras along with Time of Flight sensors to get both depth and color information while the latter method employs four Microsoft Kinect cameras to reconstruct the motion of a real world human actor. Both of these methods do not try to extract any time coherence information from the captured depth and color data.

The main motivation of our work is to present a system which can use both depth and color information and extract time coherence information from the dynamic three-dimensional content. The dynamic three-dimensional content is assumed to be in the form of a three-dimensional point cloud with color information associated with every point at every frame. We will show that we can obtain this information very easily using one or more Microsoft Kinect cameras which provide us not only the depth information of real world scene but also its color information. Our work is not limited to the data obtained by the Microsoft Kinect cameras but we will also show that our work is equally suitable for the three-dimensional content obtained using a traditional acquisition setup of multi-view color cameras. Main benefit of using Microsoft Kinect cameras is that unlike the requirement of employing eight or more color cameras only one Microsoft Kinect camera can be employed to get meaningful depth and color information of a real world dynamic scene. The main contributions of our work are:

1) *Acquisition of data using one or more Microsoft Kinect camera and organize it in a form of a three-dimensional dynamic point cloud with color information.*
2) *A new method of finding time coherent information from three-dimensional point cloud data using both color and depth information. This data can either be acquired from Microsoft Kinect cameras as described in step 1 or a traditional setup of multi-view video acquisition using color cameras.*

II. COLOR AND DEPTH VIDEOS ACQUISITION

We record a real-world actor using multiple Microsoft Kinect cameras. We employ four Microsoft Kinect cameras to capture both depth and color information of a real-world actor as shown in Fig. 1 and 2. We also make use of the data from Ahmed et al. [1] which were recorded using eight color cameras for validating our results. Our acquisition system to obtain three-dimensional dynamic point cloud with color information is only through four Microsoft Kinect cameras.

To record the moving person we place four Kinect cameras around the person each with the angle of 90 degrees with respect to their adjacent cameras. Since the Kinect projects an infra-red pattern on the real-world object to capture its depth, this can lead to interference with other cameras if multiple Kinects are recording the object at the same time. In an ideal situation the angular different between Kinects should be 180 degrees so that the pattern from one camera does not interfere with the other. In our work the maximum angular separation is 90 degrees which though causes the interference but does not negatively affects the acquisition as the depth information which is not visible from one camera is filled by one of its adjacent cameras.

The Kinect provide two data streams, one color stream and the other depth stream. Both streams have the resolution of 640x480 pixels and the frame rate of 30 frames per second. For recording the streams and minimizing any overheads the streams are captured in a memory buffer and once the recording is finished they are written to the disk. The placement of the four cameras around the objet allows capturing a dynamic scene within an area of 2m x 2.5m.

After the acquisition step we have a sequence of images of a real world person performing some motion. For each frame both the color information and the depth information is recorded. Thus we end up with 8 images for each frame of the video, two for each camera.

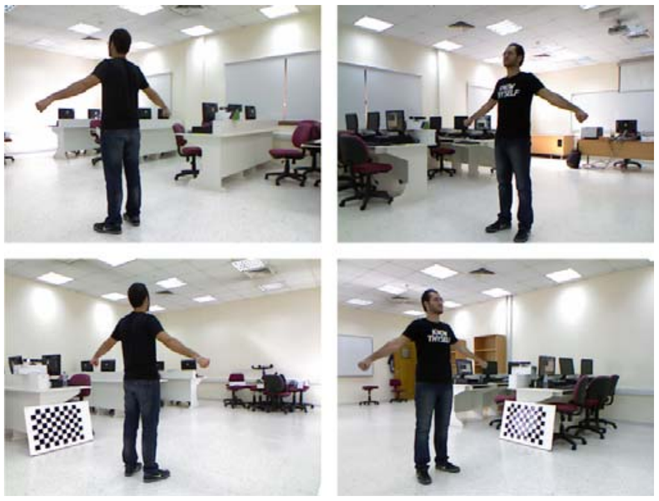
Figure 1: One color frame captured from four Kinect cameras is shown.

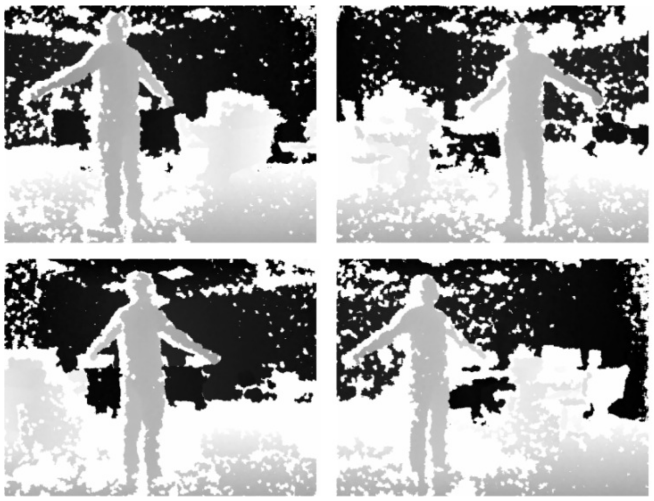
Figure 2: One depth frame captured from four Kinect cameras is shown.

CAMERA CALIBRATION AND SEGMENTATION

A Microsoft Kinect camera needs two types of camera calibrations. In each camera, there are two sensors, one depth and one color. Both sensors are needed to be calibrated individually for their intrinsic camera parameters. Similarly, an extrinsic calibration is required between the depth and color sensor to map the color information to the three-dimensional point cloud. Furthermore the depth data is represented in a relative distance from the camera which must be converted to real world meter system.

We use standard calibration method using Matlab camera calibration toolkit for the camera calibration. We record checkerboard patterns using the color sensor and the infrared sensor to find the intrinsic parameters of both sensors. Similarly same checkerboard patterns from both sensors are used to estimate the extrinsic transformation between the two sensors. This extrinsic transformation allows mapping the depth data to the color data. A point cloud and the mapping of depth data to color is shown in Fig. 3.

To convert the depth data to meters we employ the method proposed by Nicolas Burrus [4]. We use the Kinect RGB Demo software to do the full internal calibration.

Final step before getting a dynamic 3D point cloud is to segment the scene so that the real-world actor can be separated from the background. We do the background subtraction using the depth data. First the acquisition room is recorded without the human actor and later the depth information of the background is used to subtract the real-world actor from the background.

DYNAMIC 3D POINT CLOUD

The final step for getting a dynamic 3D point cloud is to merge all the cameras together in a global unified coordinate system. This global registration is an important step because without it each point cloud would be in its own coordinate frame. To achieve this global registration we first need to find out correspondences between different cameras. This is achieved by recording the checkerboard pattern at different locations for each pair of adjacent camera. The corners of the checkerboard provide the correspondences between two cameras.

Once the correspondences between all adjacent cameras are found one camera is selected as a reference camera and the correspondences are used as the starting point for the iterative closest point algorithm to find the rotation and translation transformations that maps one point cloud to the other. This transformation is found for each of two adjacent pairs and all cameras are mapped to a unified global coordinate system which is coordinate frame of the reference camera.

Global registration gives a unique pair of rotation and translation transformations for each camera and it is applied to the corresponding depth data. The final result of the global registration is a 3D point cloud for each frame of the animation. Additionally, using the mapping between color and depth cameras we also associate the color value with each point. Thus we obtain a dynamic 3D representation of a real world scene. This representation is not time coherent because each frame is independent of the other.

We also use data from Ahmed et al. [1], which have a 3D visual hull representation at every time step and a corresponding color information. We extract the point cloud from the visual hull representation and also extract the time-coherent representation of dynamic 3D content from the data as explained in the next section.

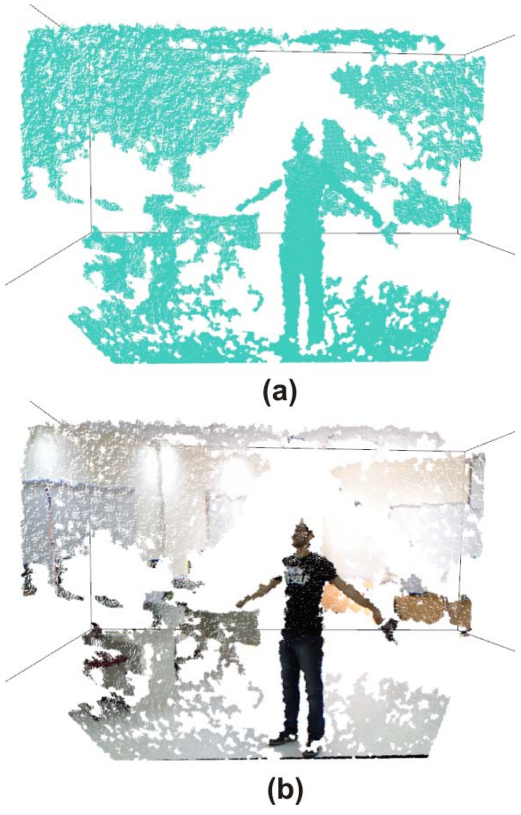

Figure 3: (a) One frame of dynamic 3D point cloud. (b) A full dynamic 3D point cloud from one camera fused with the color information.

EXTRACTING TIME COHERENCE

As explained in the previous section, the dynamic three-dimensional content obtained through either one or more Microsoft Kinects or traditional multi-view video acquisition completely lacks any temporal coherence. There is no connectivity from one point cloud to the next for each consecutive frame of the video. Thus the data is not very useful in extracting any meaningful information about the scene other than simple visualization. Even for the visualization the results do not appear good because the position of the points change so quickly from frame to frame that it distracts the viewer from the actual animation. We therefore propose a new method to extract time coherence information from this dynamic 3D point cloud using both geometric and color information. This time coherence will be found between two consecutive frames for the complete animation. Using the coherence information the point cloud at the first frame will be tracked over the whole sequence.

The first step in this method is to extract the orientation or the normal of every point of all 3D point clouds. To approximate the normal we find the normal to the plane that is tangent to the surface at that point. Since we have a point cloud there is no actual surface, rather we choose 10 nearest points to fit a plane and then find normal to that plane to find out the orientation for each point. We treat normal at each point as one of the feature of that point.

One way to match one point cloud to the next would be to just match the two points that have similar normal. This will mostly hold true as our animation is not very fast and at each frame we do not have a strong motion. But this relies on the assumption that each point has a completely unique orientation and therefore mapping from one frame to the next is trivial. In practice this is a false assumption because for every three dimensional object there are planar regions where the orientation of all the points are the same and only using the orientation information to match two 3D point clouds can never work because of the ambiguity in one to many mapping from one frame to the next.

To circumvent this ambiguity we propose to use another set of features using the color information. Since we know the mapping of the depth data to the color data each point in the point cloud has some specific color. Thus when matching two point clouds, in addition to the orientation the color information is also matched to ensure that there is no obvious incorrect mapping of the point clouds.

The color and orientation information can give us partial matching but it is still ambiguous because the actor can wear the clothes of uniform color. Therefore we introduce two more criteria of distance to make sure that a point is not mapped to another point of same color and same orientation but at farther distance. Our data has two notions of distance; one obvious notion is the three-space distance between two three-dimensional points. This can be trivially found by finding the 3D Euclidian distance between two points. The second novel notion of the distance which is one of the major contributions of our algorithm is to use SIFT to find out the feature points in the point cloud. SIFT is one of most well knows feature descriptor in image space which is invariant under transformation and varying lighting conditions. Using SIFT we find feature points in the color image and consequently using the mapping between depth and RGB images we find the feature 3D points for each frame. The SIFT matching is then used to find out the correspondences between the feature points at each frame.

Once we have feature point matches identified, we associate a distance to each point with respect to its nearest feature point. Assuming we are finding a match between two consecutive frames, e.g. 1 and 2, for all the points at frame #1 our matching function takes the following form:

$$M(x) = \alpha N(x) + \beta C(x) + \gamma F(x) + \gamma D(x) \quad (1)$$

Where x is the 3D point at the next frame, i.e. frame #2, which is used to evaluate the equation 1. M(x) is the matching distance, N(x) is the difference in orientation of x, C(x) is the difference in color of x, F(x) is the difference in feature distance of x and D(x) is the 3D Euclidean distance of x and the point at frame #1. The four parameters $\alpha, \beta, \gamma, \delta$are weighting parameters resulting in a convex combination of four terms, i.e. their sum is equal to 1 and their value is between 0 and 1. For our method we set$\alpha = 0.25, \beta = 0.2, \gamma = 0.5, \delta = 0.05$. These values are chosen because of the reliability criteria for each of the term as the feature matching is the most accurate it has maximum weight, followed by the orientation, then color and finally the 3D Euclidean distance. We choose the matching point as the one with the minimum value of the convex combination. If two points result in the same value of M(x) then point with smaller D(x) is chosen as the matching point.

RESULTS

To reconstruct three-dimensional dynamic point clouds and test our time coherence extraction method we use two types of data sets. For the first type, we recorded two sequences using four Kinect cameras where an actor is performing slow and fast motions. Each of the sequence is 250 frames long. Fig. 1 and 2 show one frame of color and depth images from all four cameras. Fig. 3 shows that we can not only capture the color and depth information using Kinects but also map them reliably to get a 3D point cloud.

We also use data from Ahmed et al. [1] to validate the time coherence extraction method. Fig. 4a shows two consecutive frames without time coherence, whereas Fig. 4b shows the same two frames with time coherence. As can be seen in Fig. 4a there is no connectivity between the two frames, e.g. feet of the actor have different shape. Using time coherence we can visualize the animation with a single 3D point cloud tracked over the sequence which can be seen in Fig. 4b. It can be observed that our method can reliably track the point cloud from one frame to the next and consequently over the course of the animation.

Our method is subject to some limitations. Most notably, we only employ one feature point in our matching function. This matching point is only the nearest matching point and it should be close enough to the point to be matched so that it approximates the geodesic distance on the surface rather than the 3D Euclidean distance. This is a very important assumption because assuming that we are tracking a point on the hand and the nearest feature point is on the leg then their distance will not be a true approximation of the geodesic distance and using this distance to match points can be erroneous if either the hand or the leg goes under some deformation. We circumvent this issue by only using the nearest feature point, which given high number of feature points does not pose any issues for our method. Ideally, one should get the geodesic distance but that is not possible given we do not have any surface data. Other possibility would be to find the body segments and use multiple feature points from the nearest segments.

Despite the limitations we show that using multiple Microsoft Kinect cameras it is possible to create dynamic 3D point clouds with the color information. We also show that given a three-dimensional content representation like this it is possible to extract time coherent information form this data and track a single 3D point cloud over the whole animation sequence.

CONCLUSION

We presented a method to extract time coherence information from a dynamic three-dimensional representation of a real-world scene. We showed that such a representation can be reconstructed using one or more Microsoft Kinect cameras. Microsoft Kinect provides both color and depth information of a scene. We combine multiple Kinect cameras and capture a complete three-dimensional dynamic scene. Our system is scalable and can be used to data obtained from any number of cameras. Our time coherence extraction method can be applied to any three-dimensional representation of the data as long it is comprised of 3D point clouds with color information. We demonstrated this by applying our method on the data obtained using only eight color cameras. In future we would like to extend our work to increase the robustness of our tracking method and explore the possibilities in the area of scene analysis and dynamic surface reconstruction.

REFERENCES

AHMED N, THEOBALT C, ROSSL C, THRUN S, and SEIDEL H.-P. Dense correspondence finding for parametrization-free animation reconstruction from video. In CVPR, 2008.

BAAK A., MULLER M., BHARAJ G., SEIDEL H.-P., THEOBALT C.: A data-driven approach for real-time full bodypose reconstruction from a depth camera. In ICCV (2011).

BERGER K., RUHL K., SCHROEDER Y., BRUEMMER C., SCHOLZ A., MAGNOR M. A.: Markerless motion captureusing multiple color-depth sensors. In VMV (2011), pp. 317–324.

BURRUS N.: Kinect rgb demo.http://nicolas.burrus.name/index.php/research/kinectrgbdemov6

CASTANEDA V., MATEUS D., NAVAB N.: Stereotime-of-flight. In ICCV (2011).

CARRANZA J., THEOBALT C., MAGNOR M. A.,SEIDEL H.-P.: Free-viewpoint video of human actors. ACMTrans. Graph. 22, 3 (2003), 569–577.

DE AGUIAR E., STOLL C., THEOBALT C., AHMEDN., SEIDEL H.-P., THRUN S.: Performance capture from sparsemulti-view video. ACM Trans. Graph. 27, 3 (2008).

GIRSHICK R., SHOTTON J., KOHLI P., CRIMINISIA., FITZGIBBON A.: Efficient regression of general-activity humanposes from depth images. In ICCV (2011).

HASLER N., ROSENHAHN B., THORMÄHLEN T.,WAND M., GALL J., SEIDEL H.-P.: Markerless motion capture with unsynchronized moving cameras. In 2009 IEEE Conference on Computer Vision and Pattern Recognition : CVPR 2009 (Miami,USA, June 2009), IEEE, pp. 224–231.

KIM Y. M., CHAN D., THEOBALT C., THRUN S.:Design and calibration of a multi-view tof sensor fusion system. In IEEE CVPR Workshop on Time-of-flight Computer Vision (Anchorage,USA, 2008), IEEE, pp. 1–7.

KIM Y. M., THEOBALT C., DIEBEL J., KOSECKA J.,MICUSIK B., THRUN S.: Multi-view image and tof sensor fusionfor dense 3d reconstruction. In IEEE Workshop on 3-D Digital Imaging and Modeling (3DIM) (Kyoto, Japan, 2009), Hilton A.,Masuda T., Shu C., (Eds.), IEEE, pp. 1542–1549.

LOWE D. G.: Object recognition from local scaleinvariant features. In ICCV (1999), pp. 1150–1157.

MICROSOFT: Kinect for microsoft windows and xbox360. http://www.kinectforwindows.org/, November 2010.

STARCK J., HILTON A.: Surface capture for performance-based animation. IEEE Computer Graphics and Applications 27, 3 (2007), 21–31.

THEOBALT C., AHMED N., ZIEGLER G., SEIDEL H.-P.: High-quality reconstruction of virtual actors from multi-view video streams. IEEE Signal Processing Magazine 24, 6 (November2007), 45–57.

VLASIC D., BARAN I., MATUSIK W., POPOVIC J.:Articulated mesh animation from multi-view silhouettes. ACMTrans. Graph. 27, 3 (2008).

WEISS A., HIRSHBERG D., BLACK M. J.: Home 3dbody scans from noisy image and range data. In ICCV (2011).

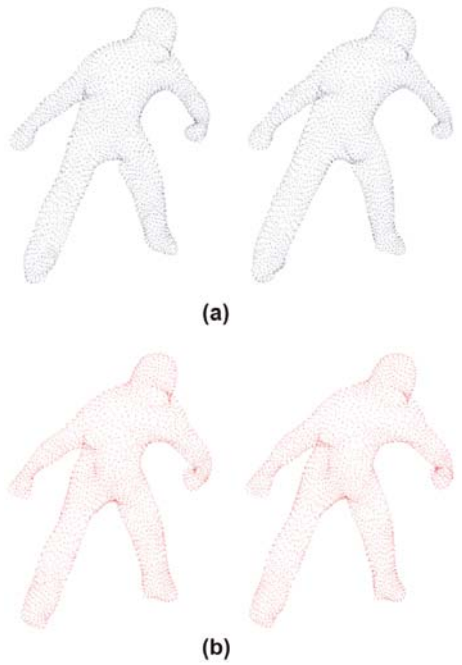

Figure 4: (a) Shows two consecutive frames from a dynamic 3D point cloud without any time coherence. (b) Show same two frames tracked using the time coherence. It can be observed, e.g. in the feet that the point cloud changes dramatically from one frame to the next without the time coherence, whereas in (b) the point cloud remains consistent.