GPU-accelerated Real-time Markerless Human Motion Capture

Christian Rau and Guido Brunnett
Computer Graphics Group, Chemnitz University of Technology, Strasse der Nationen 62, Chemnitz, Germany

Keywords: Motion Capturing, Shape-from-Silhouette, Voxelization, Pose Estimation, GPGPU.

Abstract: We present a system for capturing human motions based on video data from multiple cameras. It realizes a 3-dimensional voxel-based reconstruction of the human together with an estimation of the pose of his complete body in each frame. The use of an underlying kinematic skeleton together with an idealized geometric model can guarantee a valid pose even in the face of occlusions caused by the incomplete spatial information gained from the cameras. The data-parallel nature of the used algorithms makes them well-suited for the implementation on modern graphics hardware. In this way the motion can be captured in real-time on a single PC despite the computation of a reconstruction accurate enough for a high quality pose estimation.

1 INTRODUCTION

Most systems for accurately measuring human motion require the human to wear special markers or sensors, which may restrict his movement and can be time-consuming to attach. Furthermore, the complex technology of those systems makes them quite expensive. For these reasons there has been much research on approaches to markerless motion capturing, trying to recreate the motion based solely on the image data from one or more cameras without requiring the human to wear special equipment. But the problem of extracting a 3-dimensional motion from a set of 2-dimensional videos is a complex task often preventing the methods from performing in real-time.

But recently the programmability of graphics processors has reached a flexibility which enables them to be used for various tasks beyond just the generation of images, often outperforming CPUs by orders of magnitude for highly data-parallel computations, like those arising in markerless human motion capturing techniques. In this work we want to present such a system doing markerless motion capture on modern graphics hardware in real-time.

2 RELATED WORK

The markerless capturing of human motions has been studied extensively, a thorough overview can be gained from (Moeslund et al., 2006). The usual approach is to first extract the relevant information (the human) from the cameras' images by doing a so-called background subtraction (Toyama et al., 1999). In the next step a 3-dimensional representation of the human needs to be computed from this information. Whereas there exist approaches for the extraction of surface-based polyhedral reconstructions (Matusik et al., 2001), the usual approach is the approximation by voxelization (Cheung et al., 2000)(Caillette and Howard, 2004)(Kehl and Gool, 2006)(Corazza et al., 2010). From this reconstruction the pose of the human can be estimated. This is often done using an underlying kinematic model of the human motion system (Luck et al., 2001)(Caillette and Howard, 2004)(Kehl and Gool, 2006)(Corazza et al., 2010). Estimation can be based on various clues, be it image space information (Kehl and Gool, 2006), cluster analysis (Cheung et al., 2000)(Caillette and Howard, 2004), or anatomical knowledge about the human body (Luck et al., 2001).

In (Caillette and Howard, 2004) a hierarchical voxel grid is used to accelerate the capturing and achieve real-time performance, but a hierarchical grid can introduce a severe discretization error based on the coarseness of the base grid. The highly data-parallel nature of graphics hardware has already been used in (Hasenfratz et al., 2003) for accelerating the voxel reconstruction of a human actor from images, but hardware was not as flexible at that time, requiring the mapping of computational problems into the restricted domain of graphics processing.

Rau C. and Brunnett G..
GPU-accelerated Real-time Markerless Human Motion Capture.
DOI: 10.5220/0004282203970401
In *Proceedings of the International Conference on Computer Graphics Theory and Applications and International Conference on Information Visualization Theory and Applications* (GRAPP-2013), pages 397-401
ISBN: 978-989-8565-46-4

3 3D SHAPE RECONSTRUCTION

In a first step the relevant information, in this case the captured human, has to be extracted from the individual camera images by background subtraction. This results in a binary silhouette image with each pixel marked as either belonging to the human or to the irrelevant background. In this work we employ the technique presented in (Cheung et al., 2000).

Once the silhouette images for the current frame have been computed successfully, they are used to reconstruct a 3-dimensional representation of the human. This is achieved by the theoretical concept of the *visual hull*, the largest volume whose projection into the cameras' image planes exactly matches the corresponding silhouettes.

Due to the fact that this visual hull can have an arbitrary shape, it is usually only approximated by discretizing the 3-dimensional space into a finite grid of voxels and finding the subset of voxels that best represents the actual visual hull. This is achieved by projecting each voxel into the image planes of the individual cameras and classifying it as part of the visual hull if its projection intersects the corresponding silhouette image. For checking this intersection the projected voxel area is sampled at a small number of pixels and the whole region is classified based simply on the ratio of silhouette sample pixels to non-silhouette sample pixels, as in (Cheung et al., 2000). We further simplify the computation of the projected voxel region by representing a voxel with a disk parallel to the image plane, resulting in an easy to sample circular shape, instead of the hexagonal projection resulting from a cubic voxel.

Since we ultimately want to match the reconstructed object to a surface model of the human body (see 4), the further removal of any internal voxels, identified by having all of their respective 6-connected neighbors belonging to the foreground, is an obvious optimization step to reduce the complexity of the following steps.

4 POSE ESTIMATION

Based on the 3-dimensional reconstruction of the human his current pose is to be estimated, as represented by the joint angles of a kinematic skeleton. The use of an underlying kinematic skeleton as an abstraction of the human motion system guarantees a valid pose inside the constraints of the human body in each frame.

Unfortunately the visual hull does not carry any topological or semantic information, it need not even be connected due to errors in the background subtraction or the voxel reconstruction. The usual approach is therefore to match the visual hull to an idealized geometric model of the human body (Cheung et al., 2000)(Caillette and Howard, 2004)(Kehl and Gool, 2006)(Corazza et al., 2010). Due to the human body mainly consisting of tubular parts, the simplest geometric model to represent its shape is a set of ellipsoids (Cheung et al., 2000)(Luck et al., 2001)(Caillette and Howard, 2004), assigning each skeleton segment to a corresponding ellipsoid that describes the geometry of the surrounding body part.

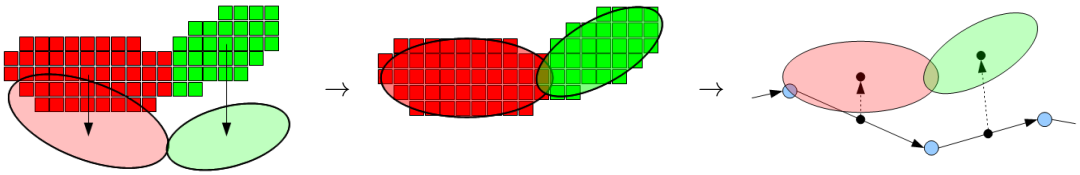
So in a first step the ellipsoid model has to realize the current pose of the reconstructed human by adapting it to his current voxel reconstruction. For this classic cluster analysis problem an *Expectation-Maximization* algorithm is a viable approach (Cheung et al., 2000)(Caillette and Howard, 2004):

1. Each voxel is assigned to the ellipsoid with the shortest distance to it, resulting in the classification of the voxels into body parts (fig. 1 (a)).
2. The ellipsoids' parameters are recomputed using a principal component analysis of the assigned voxels, resulting in the ellipsoids adapting to the voxel hull's pose (fig. 1 (b)).

Once the pose of the geometry and the locations of the individual body parts are known, the corresponding kinematic pose can be extracted therefrom. The joint angles are therefore computed using inverse kinematics, with the ellipsoids' center points defining the goals of their corresponding skeleton segments' centers (fig. 1 (c)). The iterative nature of the numeric IK methods profits from the previous frame's skeleton pose already being a good intial value for the estimation of the current frame's pose. Occlusions or errors in the background subtraction may result in ellipsoids not fitted correctly, which should not be used to drive the IK. Therefore, whenever the number of assigned voxels of an ellipsoid does not reach half the average number of assigned voxels over the whole motion, this ellipsoid does not define a goal for its corresponding segment, similar to the measure used in (Caillette and Howard, 2004).

In practice doing a complete IK over the whole skeleton turns out to be not very robust due to the high maneuverability of the root joint. Therefore, the position and orientation of the root is precomputed explicitly based on a few assumptions:

- The line connecting both hip joints always lies in a horizontal plane.
- The horizontal orientation of the root is equal to the horizontal orientation of the upper torso.
- The height of the root above the ground does not change significantly over the whole motion.



(a) expectation (b) maximization (c)

Figure 1: Ellipsoid fitting and IK-based pose estimation: (a) Assign voxels to closest ellipsoids. (b) Recompute ellipsoid parameters based on assigned voxels. (c) Ellipsoid centers as IK targets for corresponding segments.

Based on those assumptions the root orientation can be derived from the line connecting the shoulder joints as a measure for the horizontal orientation of the upper torso and from the up-axis of the torso found by a principal component analysis of all the torso's voxels. Since we don't know the shoulder joints' positions yet, the values from the previous frame are used, based on the frame-to-frame coherence. The position of the root joint can be derived from the direction of the upper legs, given by the corresponding ellipsoids, intersected with a horizontal plane at the hips' height.

During the whole pose estimation process we make heavy use of the frame-to-frame coherence by requiring the ellipsoid model's and skeleton's pose to match the human's pose of the previous frame. Of course this does not hold for the first frame. Therefore, the skeleton pose and ellipsoid parameters for the first pose have to be determined manually at the beginning. This can be simplified by requiring the human to take a certain reference pose. But in addition to a proper initialization we also need to correct the ellipsoid model each frame by readapting it to the computed kinematic pose. Otherwise the ellipsoids would tend to degenerate over the course of the motion, especially in the presence of incorrectly assigned ellipsoids.

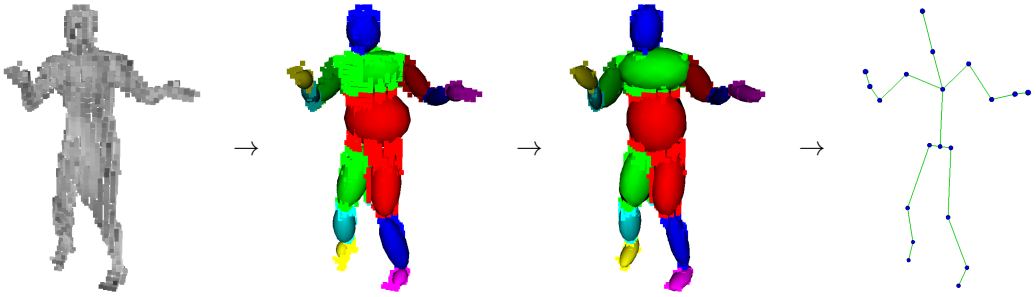
5 GPU IMPLEMENTATION

The high data-parallel nature of the used algorithms makes them well suited for being implemented on modern many-core architectures, in particular modern programmable graphics accelerators (GPUs). In this way they can be accelerated up to real-time performance even for very detailed voxel grids.

The background subtraction transforming the camera images into the binary silhouette images is a standard image processing task done independently for each pixel and requiring no synchronization between individual pixels, which makes it well suited for being implemented on the GPU. One might think about incorporating the background subtraction directly into the voxel's foreground test and thus performing it for each sample pixel of each voxel instead of each image pixel. But in practice this approach performs less efficiently, especially for larger voxel grids where the number of sample pixels is much larger than the number of image pixels, which makes the increased complexity of the voxel foreground test hide any possible gain from the omission of the background subtraction step.

The voxels' foreground test can also be invoked independently for each individual voxel. In this case we assign one thread to each voxel of the grid. This utilizes the GPU's resources sufficiently and since this thread performs a silhouette test for each individual sample pixel and each camera, its computational complexity is still high enough to hide memory latencies. The silhouette test of the sample pixels in turn profits from the GPU's texturing hardware optimized for 2-dimensional memory access. After the internal voxels have been marked as background in a following step, we finally have the boolean foreground flags of each individual voxel of the whole grid. In order for the next steps to concentrate on the relevant data only, the voxel grid needs to be compacted into a list of only the foreground voxels. This is a standard stream compaction problem often arising as part of data-parallel algorithms and can be solved efficiently by a parallel prefix sum to compute the foreground voxels' list indices (Sengupta et al., 2008), followed by a scattering step realizing the actual compaction. Since the position of a voxel is uniquely encoded in its position inside the grid and the grid resolution of each dimension does not need to be larger than 256 in practice, each foreground voxel can be compactly represented with a single 32-bit value, leaving one byte for additional state.

The expectation step of the ellipsoid fitting can again be parallelized easily without the need for synchronization by assigning a single thread to each individual foreground voxel. Since this thread computes the distance of the voxel to each ellipsoid, it again has a high computational complexity compared to a small number of memory accesses, requiring only a single load and store from/to global off-chip memory that



(a) reconstruction (b) expectation (c) maximization (d) pose estimation

Figure 2: Workflow of the motion capturing process for a single frame.

can be coalesced between multiple threads for maximum memory throughput. The ID of the nearest ellipsoid can thereby be stored compactly in the vacant byte of the 32-bit voxel state.

In order for the ellipsoids' means and covariance matrices needed for the maximization step to be accumulated efficiently, we first sort the voxels based on the IDs of their nearest ellipsoids. This can be achieved efficiently using a parallel bucket sort (Satish et al., 2009), since the number of ellipsoids is very small (15 in our case). The actual accumulation of the means and covariances can then be implemented as a segmented reduction as a modification of the segmented prefix sum, computing the data for each ellipsoid (segment). The only problem with this approach is the quite large amount of shared on-chip memory required by the individual means and covariance matrices of the voxels, which slightly reduces the number of resident threads and therefore the utilization of the GPU's resources.

Since the complexity of the final IK-based pose estimation does not depend on the number of voxels but only on the very small number of kinematic joints, it would not profit from a GPU implementation and is therefore still done on the CPU. But since it only needs information about the ellipsoids computed in the previous steps, it is not necessary to retrieve the voxel data from the GPU. And since the voxel data is computed anew each frame based on the silhouette images, the only large data that needs to be copied between CPU and GPU each frame are the cameras' images. But in practice these are usually needed on the GPU for visualizing them, anyway.

6 RESULTS

The workflow of the system for a single frame is depicted in fig. 2. To evaluate the presented motion capturing system we tested it in an artificial scenario, generated by capturing the animation of a virtual human from 4 different viewpoints and emulating a standard camera setup with an image resolution of 640×480 pixels at 30 Hz per camera. The advantage of using artificial input data is that we know the exact motion from which it was generated and can, therefore, objectively evaluate the quality of the captured motion. The used motion in this case is a 25 second dancing motion which exhibits a large range of different sub-motions. For measuring the error of the capturing, we can simply take the distance between the joint positions in the captured pose and the reference pose for each joint in each frame. This error can further be averaged over all joints and over all frames to gain an overall measure of the capturing quality, lieing between 3 and 4 cm for the tested scenario.

Table 1: Performance of the CPU and GPU based implementations for different voxel grid resolutions.

grid	32^3	64^3	128^3	256^3
CPU FPS	25	12	4	1
GPU FPS	124	121	106	49

For evaluating the influence of the voxel grid resolution on the capturing quality fig. 3 shows the average joint errors for different voxel grids plotted over the course of the whole motion. It can be seen that an increase of the voxel grid resolution results both in a smaller overall error, as well as a much smoother error curve, eliminating high-frequency errors and thus resulting in a smoother motion. This is a natural consequence of the reconstruction quality's direct dependence on the voxelization's discretization error.

The performance is shown in tab. 1 for both a CPU-based solution tested on an *Intel Core i7* with 3.4 GHz and the proposed GPU-based solution tested on an *NVIDIA GeForce GTX 580* (implemented with *OpenCL*). It can be seen that the GPU implementation realizes motion capturing in real-time up to a maximum voxel grid resolution of 256^3 voxels.

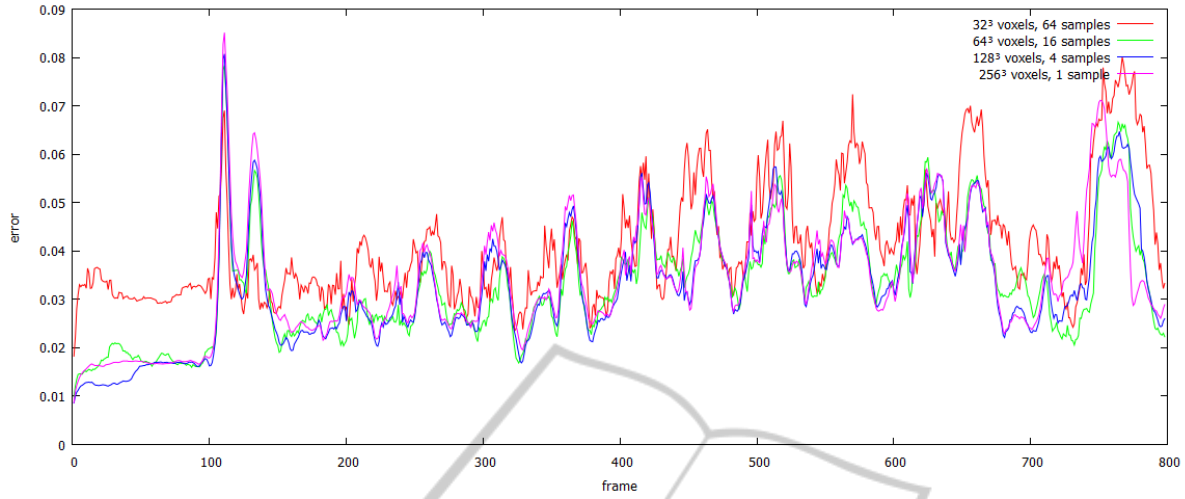


Figure 3: Average joint errors in meters over the whole motion for different voxel grid resolutions.

7 CONCLUSIONS AND FUTURE WORK

As we have seen the markerless capture of human motions is well suited for implementation on GPUs. This allows the capturing to be realized in real-time while providing a highly accurate voxel reconstruction, neccessary for a smooth and accurate motion capture.

But there is still much room for improvement of the presented system. First of all, we haven't tested it yet with real input data, which requires additional research for the background subtraction method. This was a trivial problem for the tested artificial scenario, but in reality one has to cope with illumination changes through shadows and similar problems. Furthermore, the initialization of the model for the initial pose has to be done manually, leaving room for automatization of this process. Last but not least the pose estimation makes some slightly restricting assumptions about the captured motion and still has problems with very fast motions.

REFERENCES

Caillette, F. and Howard, T. (2004). Real-time markerless human body tracking with multi-view 3-d voxel reconstruction. In *Proceedings of BMVC*, pages 597–606.

Cheung, K.-M., Kanade, T., Bouguet, J.-Y., and Holler, M. (2000). A real time system for robust 3d voxel reconstruction of human motions. In *Proceedings of the 2000 IEEE Conference on Computer Vision and Pattern Recognition (CVPR '00)*, pages 714–720.

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

Hasenfratz, J.-M., Lapierre, M., Gascuel, J.-D., and Boyer, E. (2003). Real-time capture, reconstruction and insertion into virtual world of human actors. In *Vision, Video and Graphics*, pages 49–56. Elsevier.

Kehl, R. and Gool, L. V. (2006). Markerless tracking of complex human motions from multiple views. *Computer Vision and Image Understanding*, 104(2):190–209.

Luck, J., Small, D., and Little, C. (2001). Real-time tracking of articulated human models using a 3d shape-from-silhouette method. In *Robot Vision*, Lecture Notes in Computer Science, pages 19–26. Springer Verlag.

Matusik, W., Buehler, C., and McMillan, L. (2001). Polyhedral visual hulls for real-time rendering. In *Proceedings of the 12th Eurographics Workshop on Rendering Techniques*, pages 115–126. Springer Verlag.

Moeslund, T. B., Hilton, A., and Krüger, V. (2006). A survey of advances in vision-based human motion capture and analysis. *Computer Vision and Image Understanding*, 104(2):90–126.

Satish, N., Harris, M., and Garland, M. (2009). Designing efficient sorting algorithms for manycore gpus. In *Proceedings of the 2009 IEEE International Symposium on Parallel and Distributed Processing*, pages 1–10. IEEE Computer Society.

Sengupta, S., Harris, M., and Garland, M. (2008). Efficient parallel scan algorithms for gpus. Technical report, NVIDIA.

Toyama, K., Krumm, J., Brummit, B., and Meyers, B. (1999). Wallflower: Principles and practice of background maintenance. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, pages 255–261.