GHand: A GPU algorithm for realtime hand pose estimation using depth camera

Ashwin Nanjappa, Chi Xu and Li Cheng

Bioinformatics Institute, A*STAR, Singapore

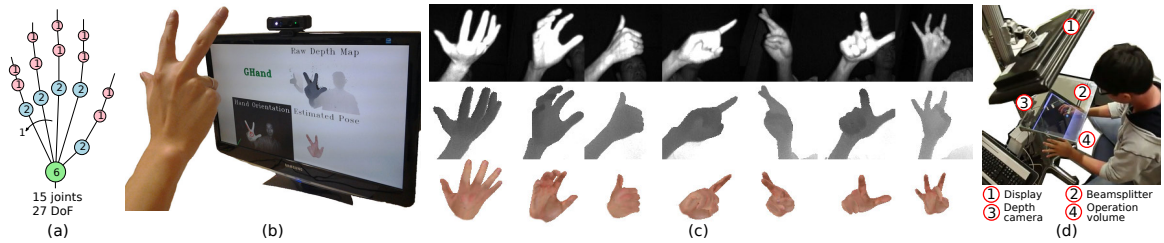


Figure 1: (a) Kinematic chain model (b) Demo of GHand (c) Results on poses with interlocking and occluded fingers (d) Demo of 3D-CuBE

Abstract

We present GHand*, a GPU algorithm for markerless hand pose estimation from a single depth image obtained from a commodity depth camera. Our method uses a dual random forest approach: the first forest estimates position and orientation of hand in 3D, while the second forest determines the joint angles of the kinematic chain of our hand model. GHand runs entirely on GPU, at a speed of 64 FPS with an average 3D joint position error of 20mm. It can detect complex poses with interlocked and occluded fingers and hidden fingertips. It requires no calibration before use, no retraining for differing hand sizes, can be used in top or front mounted setup and with moving camera.*

Categories and Subject Descriptors (according to ACM CCS): I.3.1 [Computer Graphics]: Hardware Architecture—Graphics processors I.3.6 [Computer Graphics]: Methodology and Techniques—Interaction techniques

1 Introduction

There is a growing interest in using commodity depth cameras to enable natural and intuitive interaction with computers using hand gestures. The fundamental problem here is to determine hand poses reliably and efficiently from depth images. Our GHand algorithm addresses this problem with three main contributions:

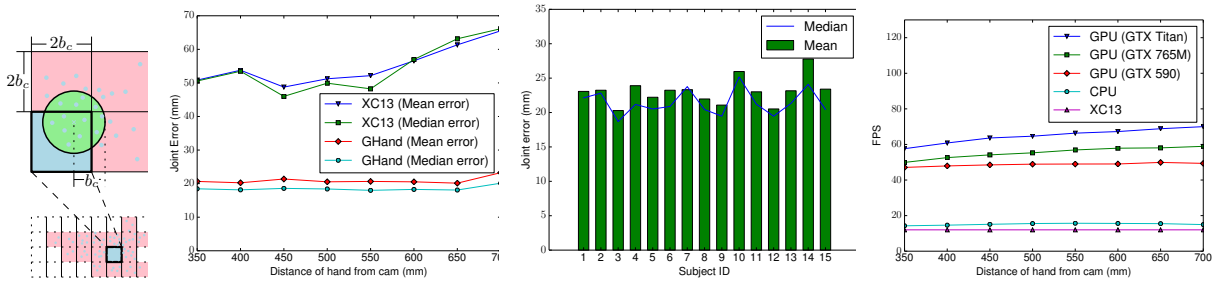
1. Estimation of the full 3D hand pose from a single depth image with high accuracy, with an average joint error of 20mm.
2. An algorithm that runs entirely on the GPU, effectively using its massive parallelism to achieve a realtime performance of 64 FPS.
3. Practical utility of GHand is demonstrated by developing a gesture-based 3D immersive workbench for biologists and by applying it to sign language recognition.

2 Algorithm

The GHand algorithm computes the hand pose from a single depth image I_t by using the approach of random forests. A kinematic chain model of the hand (Fig. 1a) with 15 joints and 27 DoF is used. During training, a synthetic database D_s of 460K hand pose depth images is created using a pose renderer that samples from a range of pose and camera parameters. Using this annotated data, a random forest F_b is trained to estimate the 3D position and orientation of the hand base using binary features. Similarly, a second random forest F_p is trained to estimate angles of all hand joints.

At runtime, I_t from the depth camera is processed to obtain the hand region. Depth values randomly sampled from I_t are applied to F_b to obtain a set of candidates, which form the votes in the voting space, in which an approximation of the mean-shift algorithm [CM02] is used to find the mode. This

DOI: 10.2312/egp.20151033



(a) (b) (c) (d)

Figure 2: (a) Illustration of GPU mean-shift in R^2 (b) Joint error for synthetic data (c) Joint error for real data (d) FPS of GHand

provides an estimate of the 3D position and orientation of the hand base to which I_t is then aligned. I_t is applied on F_p to obtain a set of possible candidates of D_s. The best candidate is chosen by using the Dominant Orientation Template (DOT) [HLI*10]. The combined result of F_b and F_p gives the complete 3D hand pose.

3 GPU optimizations

Random forests on GPU: For efficient sampling of I_t, it is mapped to 2D texture memory and texture units are configured for point filtering with unnormalized coordinates. Due to space filling curve layout of texture memory and use of texture cache, memory reads in I_t are optimized. Uncoalesced tree access is inefficient on GPU, so we arrange F_b and F_p as arrays in global memory: split and leaf nodes, tests and its parameters. Storage is reduced by arranging left and right child nodes together. To maximize parallelism, one thread per pixel is launched such that adjacent threads in a warp work on the same tree.

Mean-shift on GPU: Mean-shift is used to find mode of R^6 point cloud of votes obtained from F_b. GHand uses Epanechnikov kernel instead of Gaussian since it performs similarly but is more efficient to compute on GPU. Mean-shift is iterative and difficult to parallelize since it is costly to find points that lie within bandwidth b_c in R^6. We use an uniform grid of cells $2b_c \times 2b_c \times 2b_c$ that encloses all the points (Fig. 2a). GPU sort and parallel scan is used to map points to cells and vice versa. Centroid of every cell is computed using atomic addition and used as seeds of mean-shift. In each iteration, neighbor cells that lie in b_c are found efficiently and only its points are used.

DOT feature on GPU: For each v_{ij} obtained from F_p, training hand patches I_{ij} are collected from which the best match to I_t is found using DOT feature. I_t is randomly remapped to be robust to deformations. This is efficiently done on GPU by mapping I_t to texture memory and using bi-linear interpolation of texture units. One pixel per thread is processed on GPU to compute gradients, dominant orientation of each 8×8 block and their comparison.

4 Applications and Results

3D Computational Bioimage Explorer (3D-CuBE) We built 3D-CuBE (Fig. 1d) for 3D stereo viewing and gestural interaction of molecular structures by biologists. Structures are projected in front of user using a beamsplitter and viewed in stereo 3D using NVIDIA 3D glasses. Hand poses detected by GHand provide the gestural interaction when used with our innovative 3D menu system. This enables the user to naturally grab, rotate, zoom and pinch 3D objects. It has been demonstrated to biologists and received positive feedback.

Sign language recognition We applied GHand to gesture classification of 1 to 10 number signs of Chinese Sign Language (CSL) by using poses captured from data glove as reference. It performed well in practice with an average accuracy of 0.77.

Results Fig. 2b shows average joint error of GHand for synthetic images obtained by rendering hand poses. Compared to [XC13] we find GHand has substantially less error of 20mm, approximately the thickness of a finger. Also note that use of DOT makes GHand accuracy robust to hand distance from camera. Fig. 1c shows results of GHand on real poses. On real data captured from 15 users using Shapehand data glove, the error of GHand (Fig 2c) is low and comparable to synthetic data. Fig 2d shows that GHand uses GPU effectively to deliver realtime results (45-64 FPS) and a 5x speedup over [XC13] and CPU implementation.

5 Conclusion and Future Work

In this work, we have shown GHand to be capable of robust hand pose estimation at realtime speeds by utilizing the parallelism of GPU. The accuracy and speed of GHand make it useful as a gestural user interface which is demonstrated in applications to 3D molecular manipulation and sign language recognition. In the future we intend to use temporal information to stabilize the results of GHand.

References

[CM02] COMANICIU D., MEER P.: Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence 24* (2002), 603–619. 1

[HLI*10] HINTERSTOISSER S., LEPETIT V., ILIC S., FUA P., NAVAB N.: Dominant orientation templates for real-time detection of textureless objects. In *CVPR* (2010). 2

[XC13] XU C., CHENG L.: Efficient hand pose estimation from a single depth image. In *ICCV* (2013). 2