Skeletal Tracking using Microsoft Kinect

Abhishek Kar
Advisors: Dr. Amitabha Mukerjee & Dr. Prithwijit Guha
{akar,amit}@iitk.ac.in, prithwijit.guha@tcs.com
Department of Computer Science and Engineering, IIT Kanpur

Abstract

In this work, we attempt to tackle the problem of skeletal tracking of a human body using the Microsoft Kinect sensor. We use cues from the RGB and depth streams from the sensor to fit a stick skeleton model to the human upper body. A variety of Computer Vision techniques are used with a bottom up approach to estimate the candidate head and upper body postitions using haar-cascade detectorsa and hand positions using skin segmentation data. The data is finally integrated with the Extended Distance Transform skeletonisation algorithm to obtain a fairly accurate estimate of the the skeleton parameters. The results presented show that this method can be extended to perform in real time.

1 Introduction

Human pose estimation has been a long standing problem in computer vision with applications in Human Computer Interaction, motion capture and activity recognition. Complexities arise due to the high dimension of the search space, large number of degree of freedoms involved and the constraints involved such as no body part penetration and disallowing impossible positions. Other challenges include variation of cluttered background, body parameters, illumination changes

In this work we attempt to fit a stick skeleton model to a human body using RGB and depth streams obtained from the Microsoft Kinect sensor. We divide the paper as follows: In Section 2, we discuss some previous approaches to solving this problem. Section 3 provides a brief overview of the Kinect sensor and Section 4 describes the algorithm with brief overviews of the various steps. We give our results in Section 5 and conclude with some final remarks in Section 6.

2 Previous Work

There have been primarily two approaches to this problem - the model fitting approach and the learning approach. Model based approaches primarily involve fitting a previously formulated model onto a given image. The same problem has been modelled as an inverse kinematics problem([2] [12]) to estimate the parameters from the tracked feature points, a gradient space similarity matching problem and a maximum likelihood esmitation problem with a Markov Chain Monte Carlo approach([11]). The Achilles' heel of most of these methods is that they suffer from local extrema and depend on proper initialization.

On the other hand, the machine learning approach depends on huge amounts of data of labeled

skeletons in images. They also pay the price of computational complexity in high dimensional spaces. One of the pioneering works has been by Bourdev & Malik [3] who propose poselets to encode pose information and classify using SVMs. Ramanan and Forsythe [9] use paralleleism information to cluster appearances.

Some work has also been based on range images. Iterative Closest Point [5] has been used as a baseline approach to track a hash-initialized skeleton through subsequent frames. Kalogerakis et al. [7] classify and segment vertices in a closed 3D mesh into different parts for this purpose. Our work uses a hybrid of the two approaches using learned haar-cascade classifiers to segment head and upper body parts and then combining imformation from Extended Distance Transform and skin segmentation to fit the proposed skeleton model.

Figure 1: The Microsoft Kinect

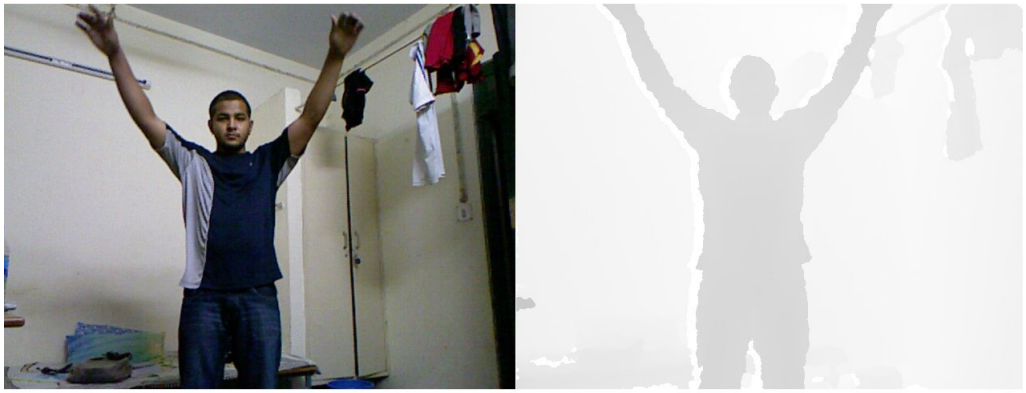

Figure 2: RGB(left) and Depth(right) images from a Kinect

3 The Microsoft Kinect sensor

The Microsoft Kinect(Figure 1) is a new product launched by Microsoft in November 2010. Its capability to produce depth and RGB streams at a price much lower than traditional range

sensors have made it a sensation in the field of Computer Vision. At the heart of the Kinect lies a time of flight camera that measures the distance of any given point from the sensor using the time taken by near-IR light to reflect from the object. In addition to it, an IR grid is projected across the scene to obtain deformation information of the grid to model surface curvature. We use the OpenKinect driver framework for the Kinect that produces 640×480 RGB and depth images(Figure 2) at 30 fps.

4 Methodology

In this work we assume that the human face and upper body are completely visible and do not have any occlusions. The Kinect sensor is static and the movement of the subject along the camera axis is constrained to a range. A block diagram of the method followed is given in Figure 3. The following sections explain each stage of the process in some detail.

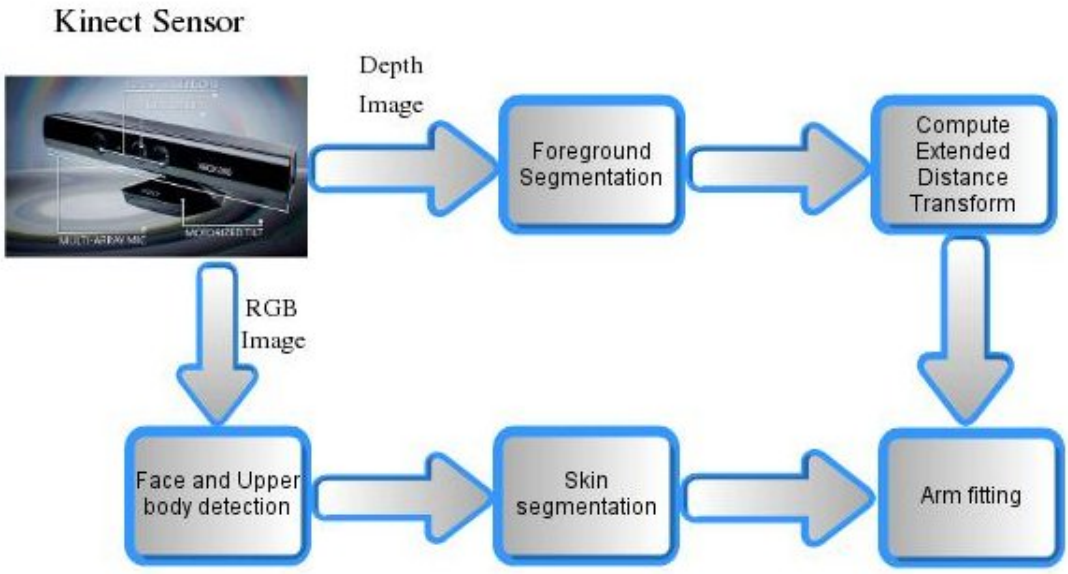

Figure 3: Overview of the algorithm

4.1 Foreground Segmentation

We use thresholding on the depth image to extract the foreground from the image. Noise is removed using morphological operations of erosion and dilation. Futher small blob removal is done to get a clean foregound segmented image. This helps us focus on only the subject in the image and calculate the Extended Distance Transform.

4.2 Haar Cascade Detection

We use the Viola-Jones [13] face and upper body detector to fiz the face and upper body positions of the subject. The detector is based on haar cascade classifiers. Each classifier uses

Figure 4: Depth thresholded image

rectagular haar features to classify the region of the image as a positive or negative match.

$$f_i = \begin{cases} +1 & v_i \geq t_i \\ -1 & v_i < t_i \end{cases} \tag{1}$$

The haar features used are shown in Figure 5

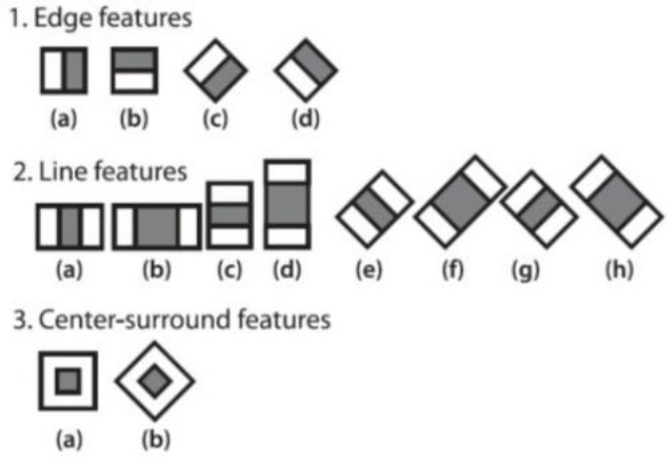

Figure 5: Haar-Features used in the Viola-Jones algorithm [13]

The detector uses AdaBoost and constructs a rejection cascade of nodes. Each node in turn is a multitree AdaBoosted classifier with a low rejection rate so that a few number of regions are classified out at each stage. A rejection at anny node terminates the whole cascade. An initial learning phase is used to learn the thresholds t_i. Boosting is then used to calculate the weights w_i. Finally the function below is used to classify whether the region is in an object of interest or not. Thus effectively a region is classified as a positive if it makes it through the whole cascade(Figure 6).

$$F = sign(w_1 f_1 + w_2 f_2 + ... + w_n f_n) \tag{2}$$

In our application, the upper body detector is run to detect the torso. The detected regions are then passed on to the face detector to detect faces. We use a frontal face detector and thus allow only a small angle of tilt for the head. The image is scaled down to half the size and the parameters for the detector are set separately for the face and the upper body detector to speed up the process.

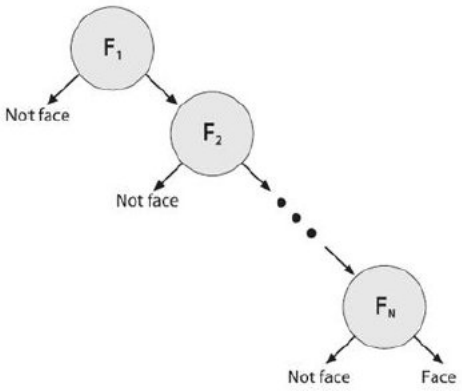

Figure 6: Rejection cascade of the haar cascade classifier [13]

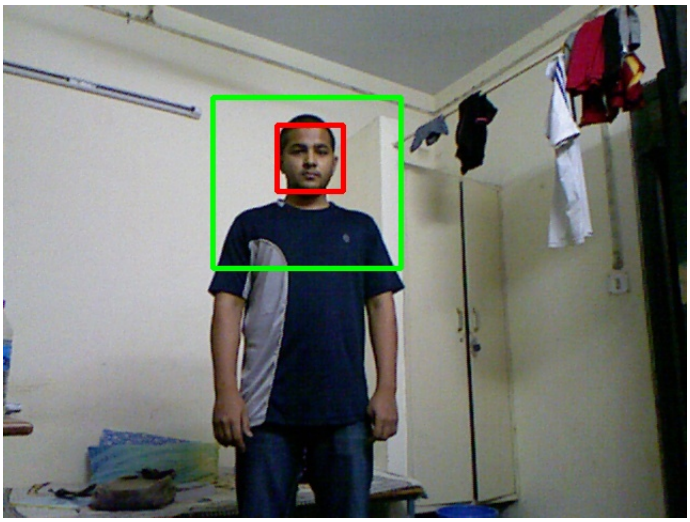

Figure 7: Detected face and upper body regions in the RGB image

4.3 The Stick Skeleton Model

We represent the human skeleton model by 7 body parts involving 8 points as shown in the Figure 9. We use anthropometric data(Figure 12) from the NASA Anthropometric Data Source Book [8] to estimate the size of the body parts. We fix the head and neck points as the centroid and mid point of the base line of the face detection rectangle. The shoulder points are fixed halfway between the face detection and torso detection rectangle base lines with the shoulder

width set as twice the face width. The lengths of the arms are fixed at 1.12 times the face width of the subject based on the anthropometric data.

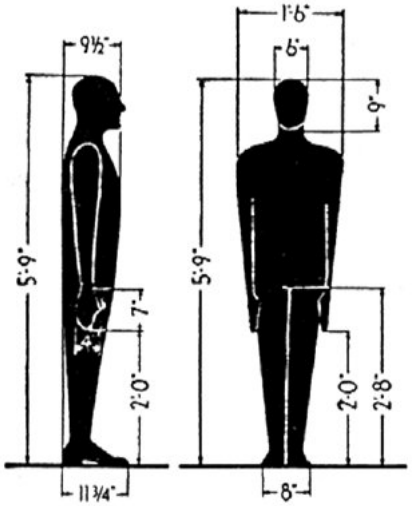

Figure 8: Anthropometric data from the NASA Anthropometric Data Source Book [8]

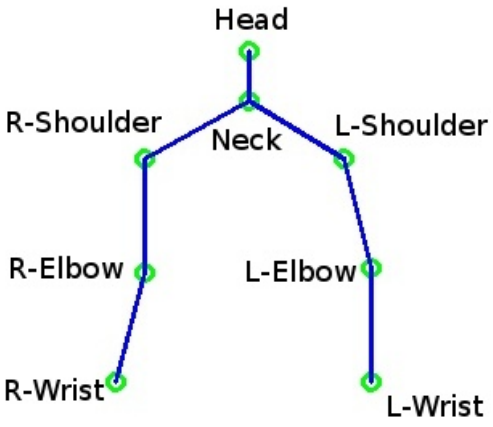

Figure 9: Stick Skeleton Model

4.4 Extended Distace Transform

The distance transform [10] on a binary image is defined as:

$$DT(p) = min\{d(p,q)|I_d(q) = 0\} \tag{3}$$

where I_d is the foreground segmented depth image. It basically transforms the image with

the pixel value at a coordinate being set as the distance to the nearest zero intensity point. The traditional distance transform uses basic horizontal, vertical shifts and knight moves. The distance used by us is the Euclidean distance $d(p,q) = \sqrt{(p_x - q_x)^2 + (p_y - q_y)^2}$ as it gives a good approximation and takes less time. The distance transform algorithm has been widely used as a skeletonisation algorithm as it basically computes a medial axis transform of the image.

As limb movements can be out of the plane, the distance transform fails to capture the whole picture. We multiply the distance transform value by a factor that takes into account the projected lengths of the limbs. We define the Extended Distance Transform for a point p wrt a reference point α as the following:

$$EDT(p,\alpha) = DT(p).(1 + \frac{|I_d(p) - I_d(\alpha)|}{I_d(\alpha)}) \quad \forall I_d(\alpha) \neq 0 \tag{4}$$

where:
DT(p) = Distance transform value at point p
I_d = Foreground segmented depth image
α = The reference point for calculation of EDT - shoulder joint for elbow point calculation and elbow joint for wrist point calculation.

Figure 10: Extended Distance Transform of a foreground segmented depth image

4.5 Skin Segmentation

We do skin colour segmentation on the foreground segmented RGB image obtained from the Kinect to aid our arm fitting process. For skin segmentation, we project the input RGB image into HSV color space and the pixels values between two thresholds hsv_{max} and hsv_{min} are set to 255 while the rest are set to 0. This gives us a binary mask with the skin regions of the subject as positive pixels. We do blob analysis on this image and remove small noisy blobs and get a clean skin mask for the image.

Figure 11: Skin segmented image

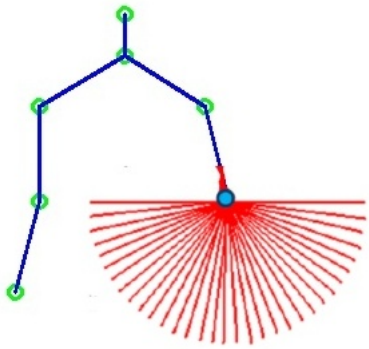

Figure 12: Illustration of the arm fitting process. The angular search space around the left elbow is shown.

4.6 Arm Fitting

In order to fit the arms we make use of the Extended Distance Transform and the skin segmentation mask computed in the previous two steps. We initiate an angular search around the pivot point (shoulder point for elbow point estimation and elbow point for wrist point estimation) at a fixed sampling frequency and compute the summed EDT values along those lines. Note that the length of the line is adjusted to the size obtained from the anthropometric ratios. We iterate over the calculated distance transformation values along the different lines calculating the skin mask value at the candidate elbow/wrist point of the line in consideration. Finally we select the direction with the maximum EDT value and end point skin mask value as 255. If no end points are found to lie in the skin region, the direction with the second largest EDT value. We discard the direction with the maximum value of EDT as it invariably corresponds to the shoulder-torso direction in case of elbow point estimation and elbow-shoulder direction in case of wrist point estimation. Temporal information is used to search in a local angular neighbourhood of the previous position of the point to speed up the computation. The sampling rate is adjustable and affects the computation time.

Figure 13: Positive results with skeleton overlayed on the RGB image

5 Results

The entire algorithm was implemented in Python using the OpenCV [4] and OpenKinect [1] libraries. Some of the results are shown in Figure 13 with the skeleton overlayed on the image. The algorithm works well on images with less occlusions. With arm occlusions (Figure 14), the distance transform map becomes a bit unreliable and the algorithm sometimes confuses between the arms for the elbow and wrist joints. As our algorithm assumes arms to be present all the time, poses with full hand occlusions do not produce correct results. A similar situation arises for occluded faces and largely occluded upper bodies. But as the detection is done separately for each frame, the algorithm is able to recover from these errors. Another problem with it is that it sometimes tends to get stuck in local extrema. Poses with projected arm lengths approaching zero, i.e. pointing poses are not well recognized (Figure 15). A rough running time analysis of the algorithm is shown in Figure 1:

Modules	**Time/frame (ms)**
Face & Upper body detection	80ms/frame
Extended Distance Transform	10ms/frame
Skin Segmentation	5ms/frame
Skeleton Fitting	8ms/frame
Total time	103ms/frame

Table 1: Running time analysis of the algorithm

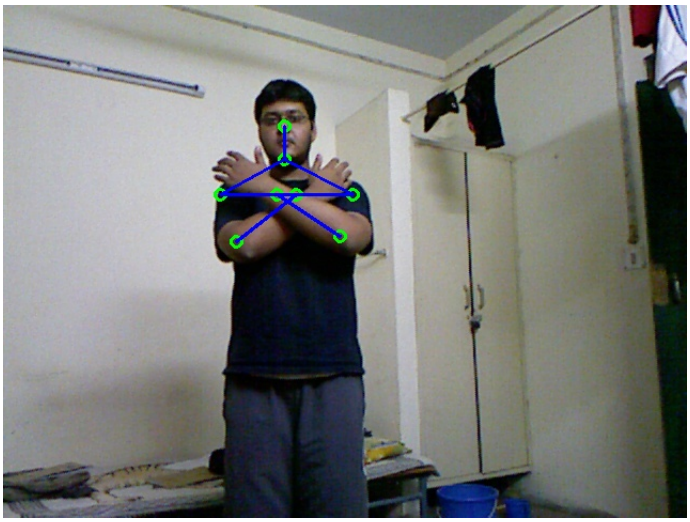

Figure 14: Incorrect skeleton with arms crossing. Confuses between the arms

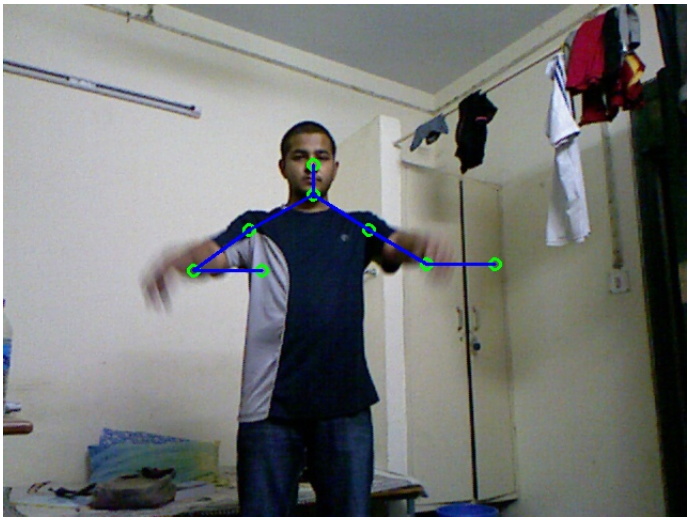

Figure 15: Incorrect skeleton with arms mainly oriented towards the camera axis

6 Conclusion

In this work, we presented a novel algorithm based on Extended Distance Transform to estimate the parameters of a stick skeleton model fitted to the upper human body. We showed a running time analysis of the algorithm and showed it to be running at about 10fps. The algorithm achieved fair accuracy in estimating non-occlusion poses. We intend to extend this work by calculating concrete accuracy figures by constructing a labeled skeleton dataset with depth images. Template matching based tracking can be used to track the faces and upper bodies through the sequence to reduce computation time. Furthermore, fore-shortening can be added to the algorithm by adjusting arm lengths on basis of the projected distance.

References

[1] Openkinect drivers.

[2] Carlos Barron and Ioannis A. Kakadiaris. Estimating anthropometry and pose from a single image. In *CVPR*, pages 1669–1676, 2000.

[3] Lubomir D. Bourdev and Jitendra Malik. Poselets: Body part detectors trained using 3d human pose annotations. In *ICCV*, pages 1365–1372, 2009.

[4] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[5] Daniel Grest, Jan Woetzel, and Reinhard Koch. Nonlinear body pose estimation from depth images. In *DAGM-Symposium*, pages 285–292, 2005.

[6] Himanshu P. Jain and Anbumani Subramanian. Real-time Upper-body Human Pose Estimation using a Depth Camera.

[7] Evangelos Kalogerakis, Aaron Hertzmann, and Karan Singh. Learning 3d mesh segmentation and labeling. *ACM Trans. Graph.*, 29(4), 2010.

[8] NASA. *Anthropometric Source Book, Vol 2.* 1978.

[9] Deva Ramanan and David A. Forsyth. Finding and tracking people from the bottom up. In *CVPR (2)*, pages 467–474, 2003.

[10] Azriel Rosenfeld and John L. Pfaltz. Distance functions on digital pictures. *Pattern Recognition*, 1(1):33–61, 1968.

[11] M. Siddiqui and G. Medioni. Human pose estimation from a single view point, real-time range sensor. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on*, pages 1 –8, june 2010.

[12] Camillo J. Taylor. Reconstruction of articulated objects from point correspondences in a single uncalibrated image. *Computer Vision and Image Understanding*, 80(3):349–363, 2000.

[13] Paul A. Viola and Michael J. Jones. Rapid object detection using a boosted cascade of simple features. In *CVPR (1)*, pages 511–518, 2001.