University of Alberta

HAND TRACKING BY FUSION OF COLOR AND A RANGE SENSOR

by

Abhishek Sen

A thesis submitted to the Faculty of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of

Master of Science

Department of Computing Science

Spring 2012
Edmonton, Alberta

Abstract

In this work we have developed a decentralized algorithm for efficient localization and tracking of hands from a sequence of depth and colour images. We deduce the location of key-points using a Bayesian framework. We use anthropomorphic constraints for modelling the interaction between body-parts. Furthermore, we incorporate an occlusion reasoning and data association preservation procedure for dealing with ambiguities. Our work is adaptive to illumination changes despite utilizing the skin-color information for tracking. Experimental results demonstrate that our system produces more accurate tracking of the head and hands in video, compared to prior research.

Acknowledgements

First and foremost I would like to thank my supervisors, Professor Anup Basu and Dr. Irene Cheng, whose encouragement, guidance and support from the initial to the final level enabled me to develop an understanding of the subject and complete this project successfully.

I offer my regards and blessings to my family and my friends who offered me endless support during this endeavour.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Gesture recognition algorithms are getting increasingly important for a wide range of applications related to Human Computer Interaction (HCI). The most intuitive gesture interaction tool is our hand, as we aim to alleviate the usage of mouse, joystick or other input devices. However the prerequisite of hand gesture recognition is tracking where the hand is localized from the scene, before higher level information is extracted from image sequence. There are two major approaches to develop hand tracking systems. These are model based approach and the appearance based approach [20]. In model based hand tracking systems [11, 25, 40, 36, 15], the image of the user's hand is fitted to a hand model and subsequent variations of the hand posture are parametrized based on the model. While this ensures accuracy in terms of tracking there are some practical issues like computational expense for model fitting. This is because hand is a highly deformable object and the parameter space for describing the model is very large. On the other hand, in appearance based tracking [30] some image features are detected and tracked over the image sequence. Ideally the chosen feature should be invariant over time so that tracking is not interrupted. One of the candidate features is skin-color because of its invariance to rotation, scale-change and shape variation.

However color-based hand tracking works well only in a controlled setup with good illumination. The illumination of a scene in an uncontrolled set-up could be less than ideal and so the skin-color detection algorithms will return poor result. Recent advances in range sensor enable us to retrieve a dense 3D scan of the environment in real-time. We augment the range data with the color image to obtain an easy localization of user and subsequent localization of hands before the tracking is initialized. The other challenges related to skin-color based tracking include the ambiguities that arise when the hands overlap with one another, or when the hand overlaps with the face. One solution is to utilize the location of the arms to disambiguate the hand locations as introduced by Buehler et al. [2] in the context of sign language recognition from video. Their approach shows an improvement in the robustness of the tracker as they successfully track the hand over long image sequence. However this approach leads to modification of the original hand tracking problem into a larger pose estimation and tracking problem, which is computationally more demanding than hand tracking. In this thesis we introduce a novel technique to reduce the computation by employing a decentralized mechanism for simultaneous detection and tracking of key-points representing the body-parts by integrating them with a directed graphical model for pose-consistency.The major advantage of our decentralized tracking scheme is that it supports parallel computation to speed up the tracking process.

Our approach to solving the pose estimation problem is a combination of *bottom-up* and *top-down* approaches. We hypothesize the location of hands, head and torso using appearance based body-part detectors in the color and range images and deduce the location of the key-points probabilistically using a Bayesian framework. We reduce the computation overhead of arm detection by predicting the location of elbow based on location of head and hand in range data. The overall methodology may be classified as a *probabilistic assemblies of part* approach [17] to solve pose estimation. We apply the skin-color detector on the foreground image to extract the hand and face region. Adaboost based face detector operator is applied to detect the face in the image and isolate the colour blob corresponding to the face. The torso is detected by using parallel line detection by probabilistic Hough transform on the range image. Since the shape of the torso is invariant, the location of shoulder joint is isolated from the torso geometrically. Now we apply the a kinematic constraint on the detected parts (face, hands and torso) to deduce the location of elbow. The key observation we use here is that if we recover the 3D coordinates of face , torso and hand we can predict the location of elbow, because of anthropomorphic ratio of human body and constraints produced by keypoints (shoulder joint, elbow joint and centroid of hand). Thus we develop the inter-part kinematic constraints by using anthropomorphic ratio and *spatial joint depth priors*. Thus, unlike the scheme employed in the *pictorial structure* based approach in [2], we don't perform any appearance based detection of upper and lower arms. This helps in reducing the dimensionality of the problem and making it computationally efficient. In addition to this we incorporate an occlusion reasoning and data association preserving mechanism in our graphical model for resolving the ambiguities that arise when skin-coloured objects move close to one another.

The thesis is organized as follows. In Chapter 2 we review some of the background and related works. In Chapter 3 we provide a brief overview of our framework and discuss the theoretical details behind the Dynamic Bayesian Network (DBN) and its decomposition and subsequent solution with particle filter. Here we also discuss the formulation of hand tracking using the DBN model. Chapter 4 discusses the implementation details of the framework. Finally, in Chapter 5 we discuss our experimental findings and show the comparison of our results with respect to state-of-the art methods. Chapter 6 talks about the overall contribution of this work and potential extensions.

Chapter 2

Related Works

In this chapter, we give a background of pose estimation literature which is relevant in context of our problem. The three aspects that we look at are:

1. Different approaches to hand tracking;
2. Different approaches to pose estimation with taxonomy;
3. Inference of Dynamic Bayesian Network in context of tracking problem.

2.1 Approaches to Hand Tracking

Hand tracking algorithms in literature can be categorized into 2 classes :

1. Model based Tracking;
2. Appearance based Tracking.

2.1.1 Model based Tracking

In model based hand tracking algorithms a model of hand is built. The image from camera is fitted against the projection of the model to estimate the model parameters. The advantage of this approach is its high accuracy. However, the downside is high computation cost for model fitting algorithm and hence not suitable for real-time applications in general.

2.1.2 Appearance based Tracking

In appearance based approaches, the hand is tracked based on features extracted from image. In this approach feature extraction is a major challenge in real world settings. The problem is more complicated in case of hand tracking because of deformable nature of the human hand, hence reliable feature extraction is mandatory.

One of the most common approaches used for hand detection in a scene is to look for skin coloured regions in the image and track them. While the idea behind using colour feature for hand tracking is simple and intuitive, there are practical challenges associated with this approach. First, skin colour is dependent on the ambient illumination conditions. There are wide range of approaches for skin colour detection in constrained environment [12]. In practical scenarios there is an issue of having skin colour variations among different users depending on complexion. Also, there are illumination issues which might result in shadows. In addition to this there may be false positive detection (non-skin object assuming skin colour).

Recent approaches in appearance based tracking use invariant features for detection of hand. Scale Invariant Feature Transform (SIFT), SURF , Histogram of Gradient (HOG) have been used in different researches to extract the features from the image. A set of training examples is used to learn the positive and negative examples of hand. A learning algorithm is then employed to classify the extracted features to be hand or non-hand.

2.2 Pose Estimation

In our work, pose estimation has been formulated as a sub-problem of tracking as it aids in tracking the hand by appearance by providing additional information. Pose estimation has been thoroughly studied over the last two decades. [32] provides some extensive reviews of various works in this area. In this section we provide a brief classification of the methodologies before we discuss some of those work which are closely related to ours.

The pose estimation problem (using a single view) is solved by using two main approaches: model-based approach, and learning-based approach. In model-based approach a geometric representation of a full human body is constructed. This representation could be 2D (eg. Cardboard model) or 3D (eg. Super-quadrics), and it encompasses the shape and kinematic constraint between various parts. A standard approach is to employ an analysis-by-synthesis methodology where by an optimization is performed to maximize the similarity between the observed images and the projection of the model [26, 34]. The learning based approaches, on the other hand, directly estimate the body pose from the appearance in the image. In this approach, the problem is solved either by learning the appearance of different body parts and detecting them independently in the image before inferring the pose probabilistically by cost function minimization (known as *probabilistic assemblies by part*) [4, 16, 23, 27, 24, 7] or by learning the mapping between the image space and model space by using a large number of example 2D views (known as *example-based approach*) [18, 29, 1, 32]. Our framework is a combination of these two paradigms that comprise the learning-based approaches. We use a part based approach to localise the hand and face from the coloured image. The relative positions of shoulder, elbow and hands are learned from training examples comprising of dense range sensor data.

There is a broad range of pose estimation algorithms in literature that are based on learning. We highlight some of the prominent works in this field. Felzenszwalb [4] introduced pictorial structure, which is a generative body model which can be applied to efficient inference of the part constellations. In [16] body parts are detected using Adaboost and are assembled together using RANSAC algorithm with prior pose constraints. In [23] Ramanan and Forsyth built a bottom-up framework where appearance of an individual is modelled by clustering candidate body segments, and then used this model to find all individuals in each frame by using a loopy belief propagation algorithm. In [27] Ren et al. have detected body parts using parallel line based detectors. They formulate the pose estimation as an integer quadratic programming problem by using the pairwise constraints between body parts. In [24] Conditional Random Fields (CRF) were used to model the parameter estimation problem and is solved by gradient ascent to maximize the conditional likelihood. In [7] the human body configuration is represented by a Markov chain where the body parts are represented by nodes of the graph parameterized using shape and location. The problem is solved using a sequential data-driven belief propagation. In [18] several exemplar manually annotated 2D views in different pose configurations are used to localize the joints in an image and subsequent pose estimation. In [29] exemplar images are efficiently mapped to pose configuration by using parameter sensitive hashing strategy for k-NN based pose retrieval. Agarwal and Triggs [1] maps the silhouettes to pose space by non-linear regression on shape descriptors recovered from silhouettes. The pose recognition framework behind Xbox Kinect [32] uses a large database (500,000 frames) of motion capture data to learn the segmentation of body parts before using depth image features with randomized decision forest classifiers to classify the user's pose. However these mapping based technique are not scalable to unconstrained environment settings because of high dimensionality of human pose space. We utilize the anthromorphic ratio of the human body to reduce the search space. Instead of learning the joint distribution of all key-points of the body we learn the position vector connecting the key-points relative to the spatial location of head. This prior based approach together with temporal constraints in DBN solves the self-occlusion problem.

In recent advances triggered by range sensors, most of the systems rely on feature extraction from depth data. These approaches are aimed at exploiting the dense correspondence between the model and the depth map and use Iterative Closest Point (ICP) to match the observation against a template. These methods are all based on local optimization and fails when the arms move to the torso. Grest et al. [6] have used non-linear least square algorithm for model fitting to edge map and established temporal correspondence from frame to frame. They have detected and tracked the upper-body pose. Knoop et al [14] uses a fusion of stereo camera and range sensor to fit a cylindrical 3D body model to the data using ICP. Siddiqui & Medioni [33] used a hand-engineered head, hand and forehand detectors to show that data driven MCMC model fitting outperforms ICP. Zhu et al. [42] uses coarse body part detectors for labelling body parts before applying kinematic constraints to estimate the location of joints from the depth-map. In [43] they modify their system for robustness when tracking failure occurs by re-detection of key-points with deformable 2D templates and using Bayesian framework for temporal prediction. However the process is computationally expensive (frame rate is 0.1 Hz). Another interesting approach is to exploit the geodesic distance on depth map to extract the human body features using an assumption that geodesic distance between any points on the body remains constant. In [5] researchers use DBN to combine generative model with a discriminative model and use the data driven evidence of body-part location, which are detected using geodesic distance. The solution is posed as MAP problem which is solved by a local hill-climbing optimization and an inference approximation procedure to generate likely states for model-based algorithm. In [28] the authors integrate these features using inverse kinematics constraints and fit skeletal model. They attempt to solve the self-occlusion problem by motion estimation using optical flow. The key-point estimation procedure in our algorithm is non-iterative and determined probabilistically using the temporal correspondence and spatial prior which solves the problems associated with deterministic approaches.

2.3 Inference of Graphical Model

Our pose estimation and hand tracking algorithm is driven by a probabilistic model. One of the standard tools for probabilistic tracking is Sequential Monte Carlo Approximation (SMCA), also known as particle filter. Application of particle filter for video tracking has been studied extensively and a large number of variants have been proposed. These variants are aimed at improving the tracking performance and also computational efficiency. These are achieved in various ways including, but not limited to, improving on the importance sampling function from which the particles are sampled, incorporation of local optimization to improve the particle likelihood [30] and using axillary particles to obtain better approximation of target location [38]. In addition, particle filter provides a principled mechanism to integrate multiple cues from image. This improves the robustness of tracker. However particle filter, in its standard form, is not suited for multi-object tracking because multiple hypothesis get fused together or tracker gets lost when occlusion event takes place. This is known as *data association problem* in object tracking literature.

There are variants of particle filter for solving the multi-target tracking problem. These algorithm explore the joint state space of the interacting targets to localize the objects. In [9] a Joint Particle Filter (JPF) has been proposed, which augments the measurement corresponding to the objects to obtain a joint observation model. The major problem associated with JPF is its computational complexity, which increases exponentially with number of targets. As a result it cannot be used in real time tracking. In [13] a Markov Chain Monte Carlo (MCMC) scheme has been introduced for multi-object tracking. They use pair-wise Markov Random Field (MRF) to model the interaction between the objects. They replace the importance sampling step in particle filter with an MCMC sampler which provides a better approximation of the hypotheses in multi-target setting.

All these methods are based on centralized scheme. Recent approaches to multi-object tracking algorithms are decentralized by nature. In [41] Yu et al. have used a collaborative tracking mechanism of multiple autonomous trackers. They have used variational analysis to solve the tracking problem which they formulated using a Markov Network. Qu et al. [22] developed a distributed scheme where the interaction between multiple objects was modelled using a *magnetic interaction potential model*, which has an intuitive resemblance with higher order Markov model. The major advantage of these distributed schemes are their parallel nature and the fact that their computational complexity increases linearly with increase in number of objects.

Probabilistic inference is often used to solve articulated object tracking which has high dimensional state space. Graphical models such as Bayesian network can be used to formulated the tracking problem in higher dimension. Sudderth et al. developed Nonparametric Belief Propagation (NBP) algorithm to solve articulated tracking problem [37]. Isard et al. proposed PAMPAS algorithm by combining ideas of belief propagation and particle filters [8]. Sigal et al. used PAMPAS for tracking human pose with loose limbed model [35]. Shen et al. [31] used Belief Propagation and Mean Field methods to infer a DBN for tracking. Wu et al. [39] proposed a computationally efficient way to solve the problem using DBN with Mean Field Monte Carlo (MFMC).

Chapter 3

Theory

3.1 Overview

In this section, we give a general description of our proposed theoretical framework, which includes tracking and occlusion reasoning in the context of hand localization. The uniqueness of the proposed graphical model is the integration of independent appearance based hand tracking with the upper body pose estimation, that can be implemented using parallel pipeline. A portion of the pipeline, concerned with tracking the left hand and elbow, is shown in Figure 3.1. The implementation details about the framework are provided later in this thesis. The intuitive idea behind the framework is reasonably simple. When the user performs a hand movement, our focus moves over to the hand, which is differentiated by its skin colour , depth and motion (visual saliency). We label the hands (left or right), depending on its connectivity to the right or left elbow. However unlike hand, elbow feature is not distinguishable from a scene at all times. To localize the elbow, we focus on the configuration of torso and hand. From depth segmentation of torso, the location of shoulders are determined, which in turn helps us locating the elbow. However when one of the hands is occluded by the other , we use the knowledge about the location of the corresponding elbow to approximate the location of the hand. We use a directed graphical model (Fig 3.2) to formalize this intuitive idea and subsequently solve the problem.

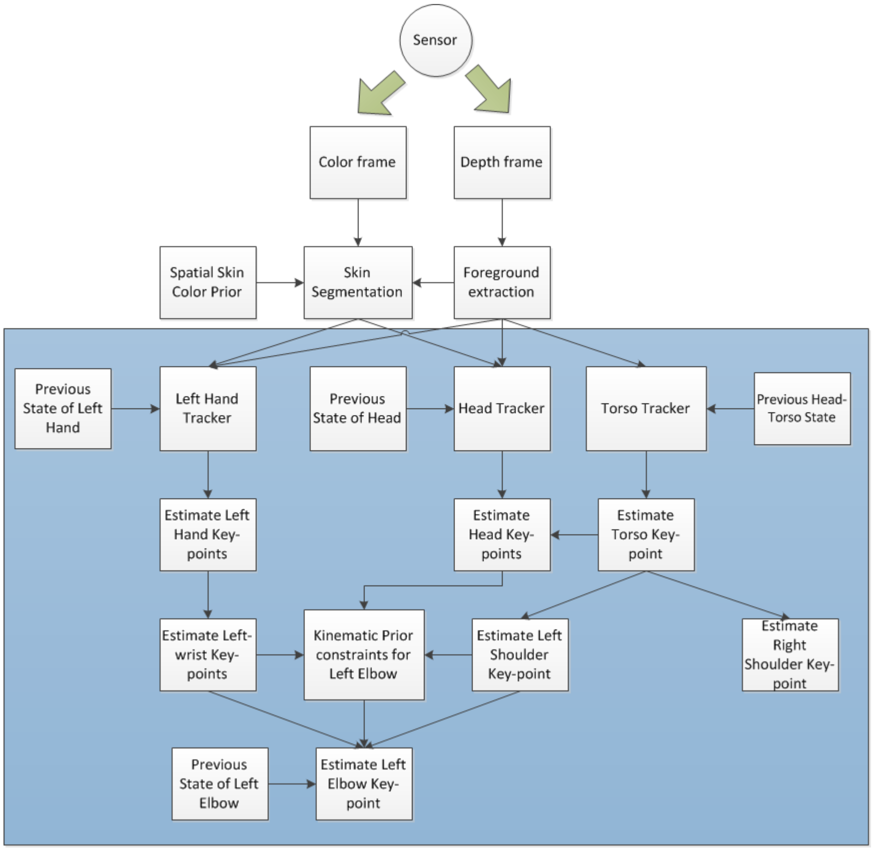

Sensor
Color frame
Depth frame
Spatial Skin Color Prior
Skin Segmentation
Foreground extraction
Previous State of Left Hand
Left Hand Tracker
Previous State of Head
Head Tracker
Torso Tracker
Previous Head-Torso State
Estimate Left Hand Key-points
Estimate Head Key-points
Estimate Torso Key-point
Estimate Left-wrist Key-points
Kinematic Prior constraints for Left Elbow
Estimate Left Shoulder Key-point
Estimate Right Shoulder Key-point
Previous State of Left Elbow
Estimate Left Elbow Key-point

Figure 3.1: Simultaneous body pose estimation and hand tracking pipeline in the presence of color and depth information. This figure shows the part of the framework that is concerned with left hand, face and left elbow tracking, in absence of occlusion. The portion of the pipeline concerned with the implementation of the Graphical Model has been highlighted in blue.

3.2 Model Description

A standard representation of articulated human body is shown in Figure 3.2. The graph can be explained as follows. Each of the key-points corresponding to various body *part* is represented by a hidden state. In our model, we perform the tracking of various body parts using corresponding key-points. We use the following symbols to describe the latent states in our graphical model.

Hand : $\mathbf{x}_{wl}$ and $\mathbf{x}_{wr}$ for left and right hands respectively.

Elbow : $\mathbf{x}_{el}$ and $\mathbf{x}_{er}$ represent the states of left and right elbow.

Shoulder : $\mathbf{x}_{sl}$ and $\mathbf{x}_{sr}$ represent the left and right shoulder.

Torso : $\mathbf{x}_T$ represents the state of torso.

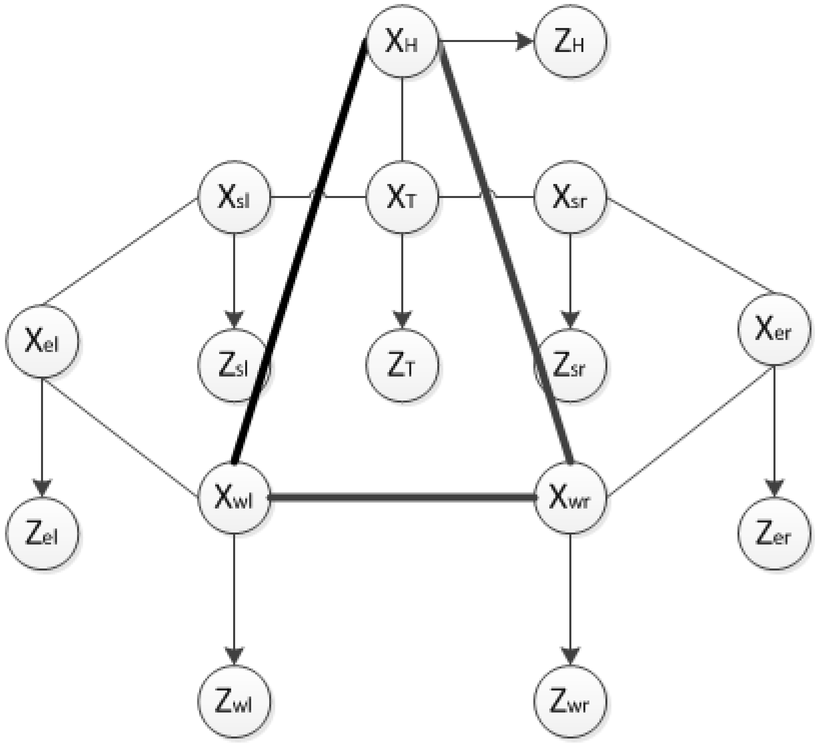


Figure 3.2: Graphical Model Representation of Human Body for pose Estimation problem. Undirected thin edges represent the inter-part constraints between key-points representing body parts. Bold edges joining a pair of nodes represent the interaction between the corresponding pair of trackers in appearance space.

Head : $\mathbf{x}_H$ represents the state of head.

Associated to each hidden state $\mathbf{x}_i$ there is an observation state $\mathbf{z}_i$. An edge joining two hidden states $\mathbf{x}_i$ and $\mathbf{x}_j$ represent the constraint between the corresponding body parts. Each of these states is parametrized as $\mathbf{x} = (cx, cy, d)$, where (cx, cy) represent the spatial coordinate for the corresponding joint and d represent the range observation of the joint. Some of the body parts are observed in coloured images. Specifically, face and hands are observed in the colour image. For each of these hidden states there is an associated color observation $\mathbf{c}_i$. We denote the state of the part at time t by $\mathbf{x}_i^t$ and the associated observation by $\mathbf{z}_i^t$. Furthermore the history of a state upto time t is $\mathbf{x}_i^{0:t}$, where $\mathbf{x}_i^0$ is the initialized state. Similarly the history of range observation upto time t is given by $\mathbf{z}_i^{0:t}$. The directed edges from $\mathbf{x}_i^{t-1}$ to $\mathbf{x}_i^t$ represent the motion dynamics of the tracker. We assume that dynamic motion model employed in our system is *Markovian* in nature.

We account for occlusion in color appearance space by introducing additional edges in the graph (bold undirected edges in Figure 3.2). These edges connect the pair of nodes associated with the objects, which potentially generates ambiguity in course of appearance based tracking. In context of appearance based hand tracking, such ambiguity arises in case of hand and face detection using skin colour. As an example, when two skin-color blobs merge it's impossible to distinguish them unless we use additional information. We attempt to resolve the ambiguity by using additional pose information. The *undirected* bold edge connecting two nodes represent the interaction energy between the objects. When this interaction energy reaches a threshold, the models corresponding to occlusion reasoning gets activated (see Figure 3.5 for the model transition diagram). There are three edges interconnecting the hidden states $\mathbf{x}_H$, $\mathbf{x}_{wl}$ and $\mathbf{x}_{wr}$ associated with skin colour regions.

The joint estimation of all the states is computationally intractable. So we decentralize the tracking problem into different components. This decentralization approach for articulated body tracking is similar to the framework used in [21]. However in their work the authors have used tracker for each body part which are conditionally dependent on the neighbourhood parts. As a result each one of their trackers is solved using iterative approach to take the neighbourhood information into account. A key observation used in our approach is that, some of the body parts are more distinguishable than the others. These parts can be tracked independently without considering their neighbourhood parts. As an example, the hands are distinguishable by skin colour and can be tracked independently without knowing the state information of elbow. Similarly, the torso can be tracked independently and efficiently using the shape information because its shape is relatively invariant. These two independent trackers are unified using state of the elbow. A graphical model in Figure 3.2 can be simplified to a directed graphical model to track the state of the elbow using its relationship with the hand and shoulder (Figure 3.3, 3.4). During an occlusion event it's reasonable to assume that independent skin colour information from hand is not reliable to estimate the location of elbow. So, the direction of the edges connecting the elbows with the corresponding hand is reversed. The undirected edges are decomposed to directed edges when each part is tracked. This idea of decomposition is similar to what has been proposed in [21]. However, the occlusion modelling in their work is dependent on the user activity (for example, walking or running). In our work this modelling is done for more random action (hand movement).The parameters of this graphical model is computed using Sequential Monte Carlo Approximation (SMCA). The interrelationship between these trackers can be represented by the layered graph representation. The layers A and C in the model correspond to the independent trackers. Layer B corresponds to the representation of elbow and is visualized as unification of the hand and torso trackers.

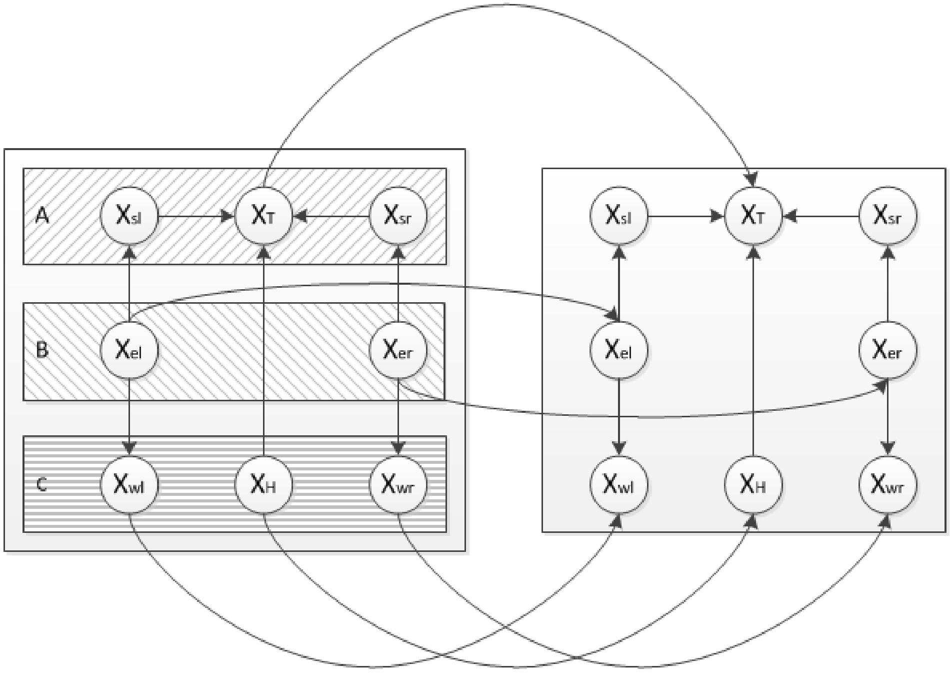


Figure 3.3: Dynamic Bayesian Network using our Layered Graphical Model representation. This image shows the temporal correspondences between successive time frames that we exploit for Sequential Monte Carlo Approximation.

3.3 Inference Algorithm

3.3.1 Graphical Model Decomposition

DBN is difficult to analyse directly because of the complex structure of the network, with several directed and undirected edges, as well as, correlation between

Table 3.1: Particle Filter Algorithm for independent color trackers

Step 1. Generate N particles, $\{\mathbf{x}_{i,t}^{(r)}\}_{r=1}^{N}$ from the dynamic motion model.

$$\mathbf{x}_{i,t}^{(r)} \sim P(\mathbf{x}_{i,t}^{(r)}|\mathbf{x}_{i,t-1}^{(r)})$$

Step 2. Compute the observation likelihood (weights) of each particle.

$$\pi_{i,t}^{(r)} = P(\mathbf{z}_{i,t}|\mathbf{x}_{i,t}^{(r)})$$

Step 3. Normalize the weights $\pi_{i,t}^{(r)}$

Step 4. Compute the expected state of the target

$$\hat{\mathbf{x}_{i,t}} = \sum_{r=1}^{N} \pi_{i,t}^{(r)} \mathbf{x}_{i,t}^{(r)}$$

Step 5. Resample the particles for the next time step. $\{\mathbf{x}_{i,t}^{(r)}, \pi_{i,t}^{(r)}\}_{r=1}^{N}$

the variables. We adopt the decentralization strategy similar to the ones reported in [21] by decomposing the network into independent components corresponding to each tracker. The decomposition is done using the following rules:

1. For each vertex in the graph retain only the vertices which are connected to the said vertex. Delete all the remaining vertices.

2. For the said vertex, retain only the edge links which are leaving the vertex. Delete the remaining edges.

The resulting subgraphs (after the decomposition operation) are Directed Acyclic independent Graphs (DAG) (see figure 3.6 for illustration). Associated to each of these DAG is a moral graph which is defined as the *”undirected graph on the same vertex set and with the same edge set obtained by including all edges in the directed graph together with all edges necessary to eliminate forbidden Wermuth configuration”* [19]. We refer the readers to [19, 21] for further details. See Figure 3.7 for moral graph construction from a DAG.

The following Markov properties are verifiable for each of the DAG. These properties are used to simplify the posterior and obtain the Sequential Monte Carlo Approximation as we describe in the following subsections.

$$
\begin{gathered}
P(\mathbf{z}_{i,t}|\mathbf{x}_{i,t},\mathbf{Z}_{Neighbour(i),t},\mathbf{z}_{i,1:t-1},\mathbf{x}_{i,1:t-1},\mathbf{Z}_{Neighbours(i),1:t-1}) = P(\mathbf{z}_{i,t}|\mathbf{x}_{i,t}) \\
P(\mathbf{x}_{i,t},\mathbf{Z}_{Neighbours(i),t}|\mathbf{x}_{i,1:t-1},\mathbf{z}_{i,1:t-1},\mathbf{Z}_{Neighbours(i),1:t-1}) = P(\mathbf{x}_{i,t},\mathbf{Z}_{Neighbours(i),t}|\mathbf{x}_{i,1:t-1}) \\
P(\mathbf{Z}_{Neighbours(i),t}|\mathbf{x}_{i,t},\mathbf{x}_{i,1:t-1}) = P(\mathbf{Z}_{Neighbours(i),t}|\mathbf{x}_{i,t}) \\
P(\mathbf{x}_{i,t}|\mathbf{x}_{i,1:t-1}) = P(\mathbf{x}_{i,t}|\mathbf{x}_{i,t-1}) \\
P(\mathbf{z}_{i,t}|\mathbf{x}_{i,t},\mathbf{x}_{j,1:t}) = P(\mathbf{z}_{i,t}|\mathbf{x}_{i,t}) \\
P(\mathbf{Z}_{Neighbours(i),t}|\mathbf{x}_{i,t}) = \prod_{j\in Neighbours(i)} P(\mathbf{z}_{j,t}|\mathbf{x}_{i,t})
\end{gathered}
\tag{3.1}
$$

In these equations, $\mathbf{Z}_{Neighbours(i),t}$ is joint observation of all neighbours of i : $\mathbf{Z}_{Neighbours(i),t} = \{\mathbf{z}_{j,t}, j \in Neighbours(i)\}$

3.3.2 Independent Trackers for Hands, Face and Torso

The independent trackers are modelled using a single chain Bayesian network. The posterior probability for this model is resolved by

$$
P(\mathbf{x}_{i,1:t}|\mathbf{c}_{i,1:t},\mathbf{z}_{i,1:t})
$$

This expression can be written as

$$
\begin{aligned}
P(\mathbf{x}_{i,1:t}|\mathbf{c}_{i,1:t},\mathbf{z}_{i,1:t}) = \eta P(\mathbf{c}_{i,t},\mathbf{z}_{i,t}|\mathbf{x}_{i,t}) \\
\int P(\mathbf{x}_{i,t}|\mathbf{x}_{i,t-1}) P(\mathbf{x}_{i,1:t-1}|\mathbf{c}_{i,1:t-1},\mathbf{z}_{i,1:t-1})\, d\mathbf{x}_{i,t-1}
\end{aligned}
\tag{3.2}
$$

The computation of the posterior probability $P(\mathbf{x}_{i,1:t}|\mathbf{c}_{i,1:t},\mathbf{z}_{i,1:t})$ at time t is thus recursively related to that at time $t-1$, $P(\mathbf{x}_{i,1:t-1}|\mathbf{c}_{i,1:t-1},\mathbf{z}_{i,1:t-1})$.

3.3.3 Tracking Elbow using Interaction with Neighbouring parts

The elbow is tracked using the directed graphical model shown in Fig 3.3. The posterior probability associated with elbow is given by

$$P(\mathbf{x}_{e,1:t}|\mathbf{z}_{e,1:t},\mathbf{z}_{s,1:t},\mathbf{z}_{w,1:t})$$

Utilizing the Markovian properties of the graphical model, this above expression can be rewritten as

$$\begin{aligned} P(\mathbf{x}_{e,1:t}|\mathbf{z}_{e,1:t},\mathbf{z}_{s,1:t},\mathbf{z}_{w,1:t}) = \\ \eta P(\mathbf{z}_{e,t}|\mathbf{x}_{e,t})(\prod_{j=s,w}\int P(\mathbf{z}_{j,t}|\mathbf{x}_{j,t})P(\mathbf{x}_{j,t}|\mathbf{x}_{e,t})\,d\mathbf{x}_{j,t}) \\ \times\int P(\mathbf{x}_{e,t}|\mathbf{x}_{e,t-1})P(\mathbf{x}_{e,1:t-1}|\mathbf{z}_{e,1:t-1},\mathbf{z}_{s,1:t-1},\mathbf{z}_{w,1:t-1})\,d\mathbf{x}_{e,t-1} \end{aligned} \tag{3.3}$$

The above equation shows the influence of adjoining key-points (shoulder joint s and centroid of hand w) in the tracking of elbow. The right hand side of the equation shows the relative constraint term $P(\mathbf{x}_{j,t}|\mathbf{x}_{e,t})$, the associated weights of shoulder and hand estimations $P(\mathbf{z}_{j,t}|\mathbf{x}_{j,t})$, the motion dynamic model of the elbow $P(\mathbf{x}_{e,t}|\mathbf{x}_{e,t-1})$ and the weight of estimated elbow state $P(\mathbf{z}_{e,t}|\mathbf{x}_{e,t})$. These terms are used to estimate the posterior at time t in a recursive manner from the posterior $P(\mathbf{x}_{e,1:t-1}|\mathbf{z}_{e,1:t-1},\mathbf{z}_{s,1:t-1},\mathbf{z}_{w,1:t-1})$, at time step $t-1$.

3.3.4 Sequential Monte Carlo Approximation

Particle filter is employed to estimate the posterior probability in equation 3.3. A set of weighted particles $S=\{(x_t^{(n)},\pi_t^{(n)})\}_{n=1}^N$ is used to represent the posterior distribution $P(\mathbf{x}_{i,1:t}|\mathbf{c}_{i,1:t},\mathbf{z}_{i,1:t})$, where the weight of the n^{th} particle x_t^n at time t is given by $\pi_t^{(n)}$. Approximating the posterior probability distribution using these particles, we have

$$P(\mathbf{x}_{i,1:t}|\mathbf{z}_{i,1:t}) \approx \sum_n \pi_{t-1}^n P(\mathbf{x}_{i,t}|\mathbf{x}_{i,t-1}) \tag{3.4}$$

The sample set evolution is modelled using the propagation of the particles according to a dynamic model. In the above equation the model is governed by the pdf $P(\mathbf{x}_t|\mathbf{x}_{t-1})$.

We use the principle of *Sequential Importance Sampling* to choose the weights of the particles. The samples $(\mathbf{x}_{i,t}^{k})$ can be generated from an importance density $Q(\cdot)$ with associated importance weights

$$\pi_{i,t}^{(n)} \propto \frac{P(\mathbf{x}_{i,1:t}|\mathbf{z}_{i,1:t})}{Q(\cdot)} \tag{3.5}$$

For trackers of key-points (for example, elbow), which have interaction with neighbouring components, we have

$$\pi_{i,t}^{(n)} \propto \frac{P(\mathbf{x}_{i,1:t}|\mathbf{z}_{i,1:t}, \mathbf{Z}_{Neighbour(i),1:t})}{Q(\cdot)} \tag{3.6}$$

For the independent trackers, if the importance density is chosen as $Q(\mathbf{x}_{i,1:t}|\mathbf{z}_{i,1:t})$, then

$$\pi_{i,t}^{(n)} \propto \frac{P(\mathbf{x}_{i,1:t}|\mathbf{z}_{i,1:t})}{Q(\mathbf{x}_{i,1:t}|\mathbf{z}_{i,1:t})} \tag{3.7}$$

If the importance density is chosen to factorize

$$Q(\mathbf{x}_{i,1:t}|\mathbf{z}_{i,1:t}) = Q(\mathbf{x}_{i,t}|\mathbf{x}_{i,1:t-1}, \mathbf{z}_{i,1:t})Q(\mathbf{x}_{i,1:t-1}|\mathbf{z}_{i,1:t-1}) \tag{3.8}$$

By substitution we have

$$\pi_{i,t}^{(n)} = \pi_{i,t-1}^{(n)} \frac{P(\mathbf{z}_{i,t}|\mathbf{x}_{i,t}^{n})P(\mathbf{x}_{i,t}^{n}|\mathbf{x}_{i,t-1}^{n})}{Q(\mathbf{x}_{i,t}^{n}|\mathbf{x}_{i,1:t-1}^{n}, \mathbf{z}_{i,1:t})} \tag{3.9}$$

Furthermore, if $Q(\mathbf{x}_{i,t}|\mathbf{x}_{i,1:t-1}, \mathbf{z}_{i,1:t}) = Q(\mathbf{x}_{i,t}|\mathbf{x}_{i,t-1}, \mathbf{z}_{i,t})$, then the importance density becomes dependent on $\mathbf{x}_{i,t-1}$ and $\mathbf{z}_{i,t}$. The modified weight becomes

$$\pi_{i,t}^{(n)} = \pi_{i,t-1}^{(n)} \frac{P(\mathbf{z}_{i,t}|\mathbf{x}_{i,t}^{n})P(\mathbf{x}_{i,t}^{n}|\mathbf{x}_{i,t-1}^{n})}{Q(\mathbf{x}_{i,t}|\mathbf{x}_{i,t-1}, \mathbf{z}_{i,t})} \tag{3.10}$$

It can be shown that these weights give an approximation of the posterior density for the independent tracker. For the trackers having interaction with neighbouring trackers, the importance density is chosen to factorize such that

$$\begin{aligned} Q(\mathbf{x}_{i,1:t}|\mathbf{z}_{i,1:t}, \mathbf{Z}_{Neighbours(i),1:t}) = Q(\mathbf{x}_{i,t}|\mathbf{x}_{i,1:t-1}, \mathbf{z}_{i,1:t}, \mathbf{Z}_{Neighbours(i),1:t}) \\ Q(\mathbf{x}_{i,1:t-1}|\mathbf{z}_{i,1:t-1}, \mathbf{Z}_{Neighbours(i),1:t}) \end{aligned} \tag{3.11}$$

Substituting this in the weight expression we have

$$\pi_{i,t}^{(n)} = \pi_{i,t-1}^{(n)} \frac{P(\mathbf{z}_{i,t}|\mathbf{x}_{i,t}^{n})P(\mathbf{x}_{i,t}^{n}|\mathbf{x}_{i,t-1}^{n})}{Q(\mathbf{x}_{i,t}^{n}|\mathbf{x}_{i,1:t-1}^{n}, \mathbf{z}_{i,1:t}, \mathbf{Z}_{Neighbours(i),1:t})} \times \prod_{j \in Neighbours(i)} \{\sum_{l=1}^{N_j^s} P(\mathbf{z}_{j,t}|\mathbf{x}_{j,t}^{l})P(\mathbf{x}_{j,t}^{l}|\mathbf{x}_{i,t}^{n})\} \quad (3.12)$$

In this equation the density function $P(\mathbf{x}_{j,t}^{l}|\mathbf{x}_{i,t}^{n})$ models the interaction between samples of part i and its neighbour, part j. $P(\mathbf{z}_{j,t}|\mathbf{x}_{j,t}^{l})$ is the weighting bias that is added to this interaction.

3.3.5 Inter-part Interaction using Spatial-Kinematic Prior

In this section we define the interaction probability density that we use to compute the posterior equations defined in the previous sections. This density function $P(\mathbf{x}_{j,t}|\mathbf{x}_{i,t})$ is used to quantify the interaction between the part $\mathbf{x}_i$ and the neighbouring part $\mathbf{x}_j$. As an example, in Figure 3.3 the directed edge connecting the nodes $\mathbf{x}_{el}$ and $\mathbf{x}_{wl}$ represents the interaction (spatial and kinematic constraints) between the corresponding key-points. The advantage of this spatial prior is that it incorporates the physical constraints between various joints effectively without trying to parametrize the joint interrelationship using Gaussian distribution [21] or mixture of Gaussians [35]. We discuss the implementation of prior computation in section 4.9.

3.4 Decentralized Occlusion Reasoning

Occlusion reasoning is imperative for consistent tracking over a long period of time in multi-hypothesis setting. Occlusion reasoning is specifically important in our problem setting because of the labelling ambiguity created by skin-color in terms of detecting the hand and face in the image. While the targets are reasonably far from one another, this labelling can be achieved by using the strategy described in the previous section. However, when the targets move closer, a more sophisticated reasoning strategy needs to be used.

3.4.1 Modelling the Multi-object Interaction

Each of the skin colour object have an *interaction zone*. When another skin colour object enters that zone, the graphical model for occlusion reasoning gets activated (Figure 3.5). The inference sub-problem is represented as $P(\mathbf{x}_{i,1:t}|\mathbf{z}_{i,1:t}, \mathbf{z}_{j,1:t}, \mathbf{z}_{k,1:t})$, where $\mathbf{x}_{i,1:t}$ represent the state of the object in question, $\mathbf{x}_{k,1:t}$ is the state of the object which is interfering in tracking and $\mathbf{x}_{j,1:t}$ is the state of the neighbouring body part which provides the auxiliary information to improve the performance of tracking. Assuming that the graphical model is *Markovian*, one can simplify the posterior as follow.

$$\begin{aligned}
&P(\mathbf{x}_{i,1:t}|\mathbf{z}_{i,1:t}, \mathbf{z}_{j,1:t}, \mathbf{z}_{k,1:t}) = \\
&\eta P(\mathbf{z}_{i,t}|\mathbf{x}_{i,t}) \int P(\mathbf{z}_{j,t}|\mathbf{x}_{j,t}) P(\mathbf{x}_{j,t}|\mathbf{x}_{i,t})\, d\mathbf{x}_{j,t} \times \int P(\mathbf{z}_{k,t}|\mathbf{x}_{k,t}) P(\mathbf{x}_{k,t}|\mathbf{x}_{i,t}) d\mathbf{x}_{k,t} \\
&\qquad \times \int P(\mathbf{x}_{i,t}|\mathbf{x}_{i,t-1}) P(\mathbf{x}_{i,1:t-1}|\mathbf{z}_{i,1:t-1}, \mathbf{z}_{j,1:t-1}, \mathbf{z}_{k,1:t-1})\, d\mathbf{x}_{i,t-1} \qquad (3.13)
\end{aligned}$$

The right hand of the above equation shows three components. The integral $\int P(\mathbf{z}_{j,t}|\mathbf{x}_{j,t}) P(\mathbf{x}_{j,t}|\mathbf{x}_{i,t})\, d\mathbf{x}_{j,t}$ describes the interaction between the object and the neighbouring body part. For example, the interaction between the left hand and left elbow is modelled using this integral. The second integral of the equation, $\int P(\mathbf{z}_{k,t}|\mathbf{x}_{k,t}) P(\mathbf{x}_{k,t}|\mathbf{x}_{i,t}) d\mathbf{x}_{k,t}$ models the occlusion event created by the interaction between the hypotheses $\mathbf{x}_i$ and $\mathbf{x}_k$ at time t. The third component in the right hand side represents the temporal coherence.

3.4.2 Inter-tracker Interaction Densities

There are two types of hidden state interaction priors required to solve this posterior. The prior used to describe the interaction between neighbouring parts has been described in an earlier section. The other interaction density function, $P(\mathbf{x}_{k,t}|\mathbf{x}_{i,t})$, used in the above equation is the *inter-tracker interaction density* (shown in Figure 3.2 and 3.4 using bold edges). We track the skin color hand blobs using bounding rectangles. When two blobs $\mathbf{x}_{i,t}$ and $\mathbf{x}_{k,t}$ are close to one another, three different scenarios are possible

1. Object $\mathbf{x}_{k,t}$ occludes Object $\mathbf{x}_{i,t}$
2. Object $\mathbf{x}_{k,t}$ is together with Object $\mathbf{x}_{i,t}$
3. Object $\mathbf{x}_{i,t}$ occludes Object $\mathbf{x}_{k,t}$

We model the resulting distribution as follows :

$$P(\mathbf{x}_{k,t}|\mathbf{x}_{i,t}) = \phi_{i,k}(1 - \mathcal{N}(-d(\mathbf{x}_{k,t}, \mathbf{x}_{i,t}); 0, \sigma_{i,k})) \tag{3.14}$$

where $d(\mathbf{x}_{k,t}, \mathbf{x}_{i,t})$ represents the difference in depth between the predicted keypoint locations. The factor $\phi_{i,k}$ is a depth ordering factor, which rewards the hypothesis pair if they preserve the depth ordering from the previous frame. We define $\phi_{i,k}$ as :

$$\phi_{i,k} = 0.5(1 + correctness * \mathcal{N}(-\delta_{depth}(\mathbf{x}_{k,t}, \mathbf{x}_{i,t})) \tag{3.15}$$

where, $correctness$ is 1, if the depth order is preserved and -1 if it is not.

Table 3.2: Particle Filter Algorithm with Neighbour Interaction (no Occlusion)

Step 1. Generate the particles, $\{\mathbf{x}_i^{(r)}\}_{r=1}^{N_i}$, from the importance sampling density.

$$\mathbf{x}_{i,t}^{(r)} \sim Q(\mathbf{x}_{i,t}|\mathbf{x}_{i,t-1}, \mathbf{z}_{i,t}, \mathbf{Z}_{Neighbours(i),t})$$

Step 2. Compute the likelihood (weights) $\pi_{i,t}^{(r)}$ of each particle using Equation 3.12.

Step 3. Normalize the weights $\pi_{i,t}^{(r)}$, so that $\sum_{r=1}^{N_i} \pi_{i,t}^{(r)} = 1$

Step 4. Compute the expected state of the target

$$\hat{\mathbf{x}_{i,t}} = \sum_{r=1}^{n} \pi_{i,t}^{(r)} \mathbf{x}_{i,t}^{(r)}$$

Step 5. Resample the particles for the next time step. $\{\mathbf{x}_{i,t}^{(r)}, \pi_{i,t}^{(r)}\}$

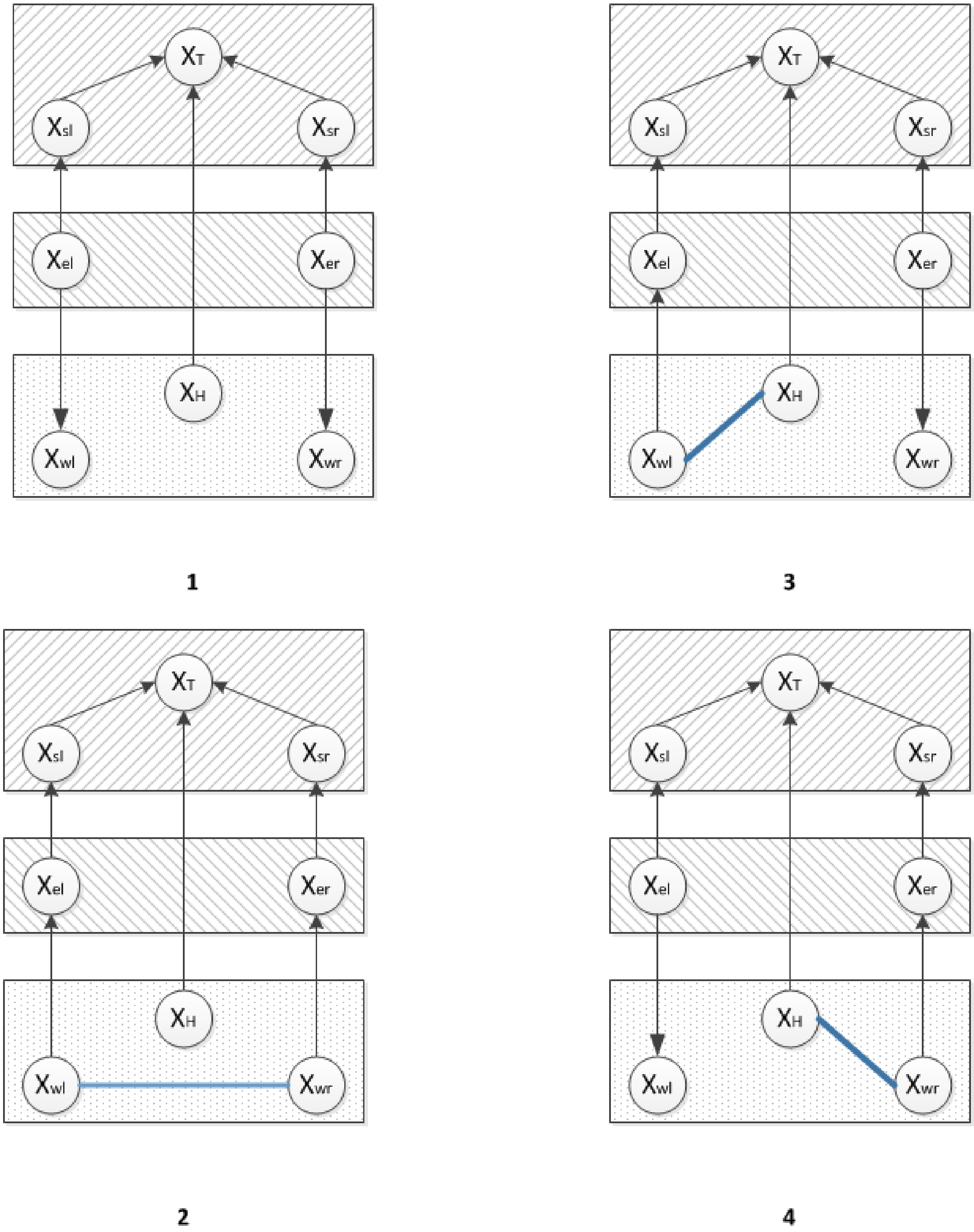


Figure 3.4: Variations of Layered Graphical Model representation for occlusion reasoning in colour space. (1) represents the scenario when the skin coloured object are not interacting with each other during tracking. (2) when left hand & right hand are close, (3) when the left hand & face are interacting with one another and (4) when the right hand & face are interacting. For models 2, 3 and 4 the undirected bold edge represents the inter-tracker interaction density.

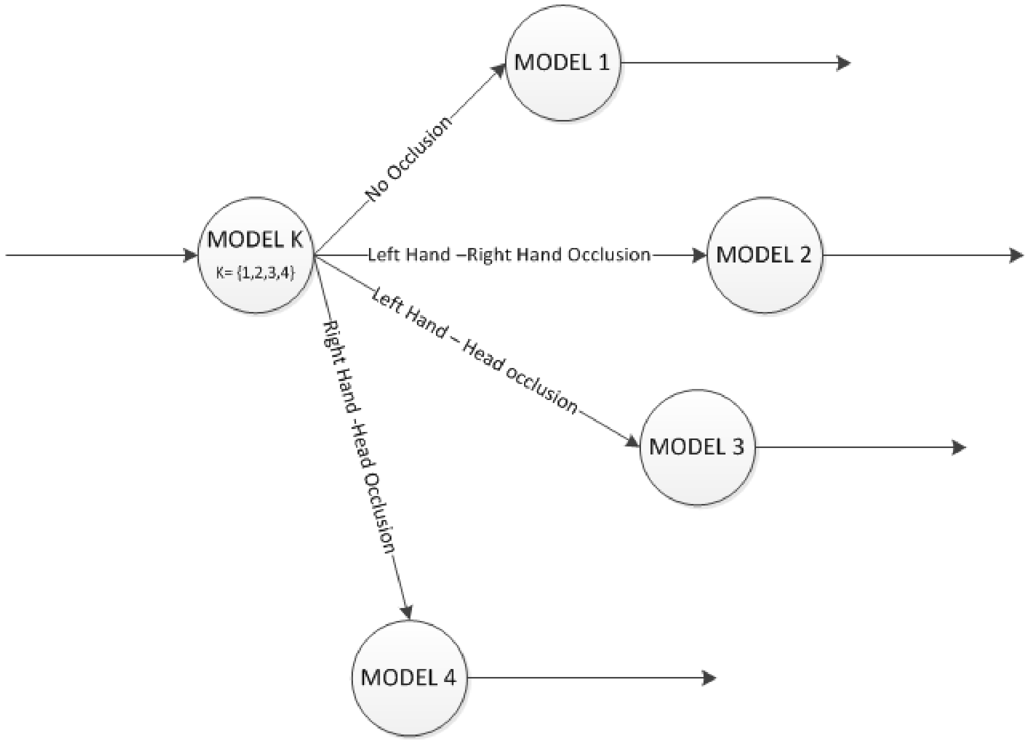

MODEL K
K= {1,2,3,4}
No Occlusion
MODEL 1
Left Hand –Right Hand Occlusion
MODEL 2
Left Hand – Head occlusion
MODEL 3
Right Hand -Head Occlusion
MODEL 4

Figure 3.5: Finite State Machine representation of model switching for an occlusion event.

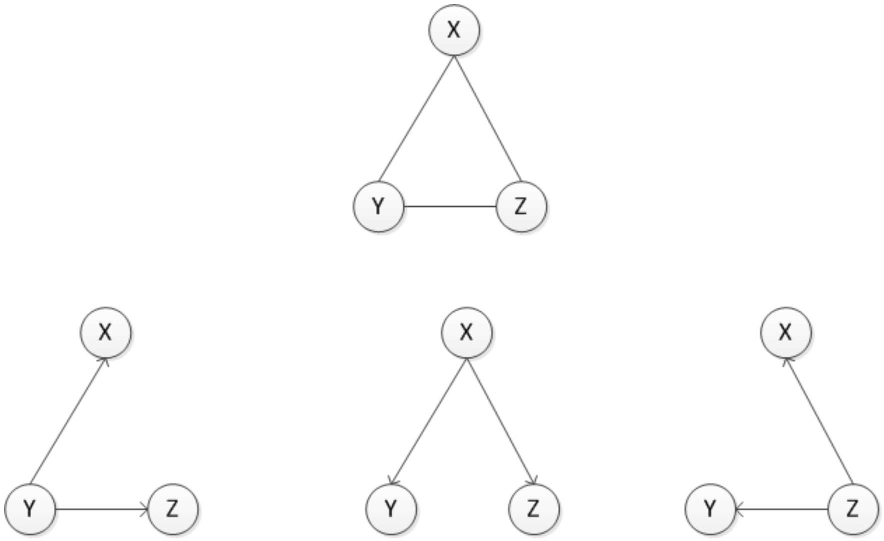


Figure 3.6: Decomposition of undirected graphical model into Directed Acyclic independent Subgraphs.

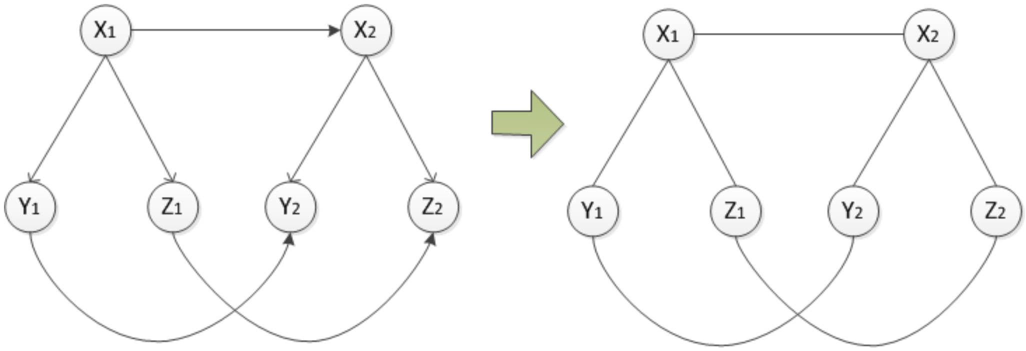


Figure 3.7: A Directed Acyclic Subgraph(left) and the corresponding Moral Graph(right)

Table 3.3: Particle Filter Algorithm with Neighbour Interaction (under Occlusion)

```
Step 1. Generate particles $\{\mathbf{x}_i^{(r)}\}_{r=1}^{N_i}$from the Importance sampling density.
        $\mathbf{x}_{i,t}^{(r)} \sim Q(\mathbf{x}_{i,t}|\mathbf{x}_{i,t-1}, \mathbf{z}_{i,t}, \mathbf{Z}_{Neighbours(i),t}, \mathbf{Z}_{Trackers(i),t})$
Step 2. Compute the likelihood (weights) $\pi_{i,t}^{(r)}$ of each particle using Equation 3.12.
Step 3. Normalize the weights $\pi_{i,t}^{(r)}$, so that $\sum_{r=1}^{N_i} \pi_{i,t}^{(r)} = 1$
Step 4. Compute the temporary expected state of the target
        $\hat{\mathbf{x}_{i,t}} = \sum_n \pi_{i,t}^{(r)} \mathbf{x}_{i,t}^{(r)}$
For iter = 1:maxIter
    Step 5. Update weights
            $\pi_{i,t}^{(r)} = \pi_{i,t}^{(r)} * \prod_{k \in Trackers(i)} P(\mathbf{x}_{k,t}|\mathbf{x}_{i,t}^{(r)})$
    Step 6. Normalize weights $\pi_{i,t}^{(r)}$
    Step 7. Estimate
            $\hat{\mathbf{x}_{i,t}} = \sum_n \pi_{i,t}^{(r)} \mathbf{x}_{i,t}^{(r)}$
    /*estimating each of the interactive trackers. M is the number of trackers interacting with $i$ */
    For k = 1:M
        Step 8. Update weights
                $\pi_{k,t}^{(r)} = P(\mathbf{z}_{k,t}|\mathbf{x}_{k,t}^{(r)}) \prod_{q \in Trackers(k)} P(\mathbf{x}_{q,t}|\mathbf{x}_{k,t}^{(r)})$
        Step 9. Normalize weights $\pi_{k,t}^{(r)}$
        Step 10. Estimate
                $\hat{\mathbf{x}_{k,t}} = \sum_n \pi_{k,t}^{(r)} \mathbf{x}_{k,t}^{(r)}$
    end
    Step 11. Resample the particles for the next iteration. $\{\mathbf{x}_{i,t}^{(r)}, \pi_{i,t}^{(r)}\}$ , $\{\mathbf{x}_{k,t}^{(r)}, \pi_{k,t}^{(r)}\}$
end
```

Chapter 4

Implementation Overview

In our tracking framework, we assume that the depth camera can capture the upper-body of the user completely. The user must face the camera and there must not be any occlusion during initialization. In the beginning of the tracking phase the user must stand straight with two arms outstretched and the palm facing the camera. This is used to initialize the locations of various body parts. Our system is invariant to illumination change as long as user's face is visible to the color camera.

We make the following assumptions at the *current stage of the implementation*:

1. The user stays upfront at all time in front of the camera, although he can turn his head once the tracker has been initialized.

2. The user is constrained to stand at the same location, where he was standing, when tracking initialized.

We show a portion of the pipeline of the tracking framework in Fig 3.1. The following subsection explains various components of the pipeline.

4.1 Depth Camera

Depth imaging technology has advanced dramatically over the last few years. Very recently Microsoft developed the Kinect sensor, there by making the technology available at a consumer-affordable price. The device projects a *structured infrared pattern* on the scene and process the deformation of the reflected pattern to estimate a depth map of the scene. We use the Kinect camera which gives 640 x 480 image at 30 fps with a depth resolution of a few centimetres.

Depth cameras offer several advantages over the traditional intensity sensors, working in low light levels, giving calibrated scale estimate, being color and texture invariant and resolving silhouette ambiguity of pose. They also greatly simplify the task of background subtraction.

4.2 Foreground Segmentation

The foreground is extracted by applying threshold on the depth information obtained from the Kinect. The blobs are checked for false positives by applying face detection algorithm on the corresponding image region of the coloured image. The illustration of foreground extraction is shown in Figure 4.1.

4.3 Head-Neck-Torso Segmentation

Prior knowledge on the relative size of body parts is used when building a model for segmenting the head and torso. In the initial phase of tracking, the user is constrained to stand at a specific pose, so that the algorithm processes the depth map. We use a template-matching approach for extracting the head and torso from the depth map. The template is initialized based on detection of *nearly* vertical lines using Hough Transform when the system is initialized. The horizontal edges of the templates are initialized based on location of the head. Figure 4.1 shows an illustration of template initialization.

4.4 Hand-Tracking Initialization

For initialization, the user stands with his hand stretched horizontally on both directions. This serves two purpose : first, the parameters which describe the user's body-parts can be estimated by estimating the relative length of upper and lower arms. Second, our method can utilize the observation that the extremity of the arms represents the user's hand. If the arms are not occluding the torso or head, subtraction of torso and head from the image leads to the localization of the arms. Otherwise depth thresholding is applied relative to the torso. We track the hand using

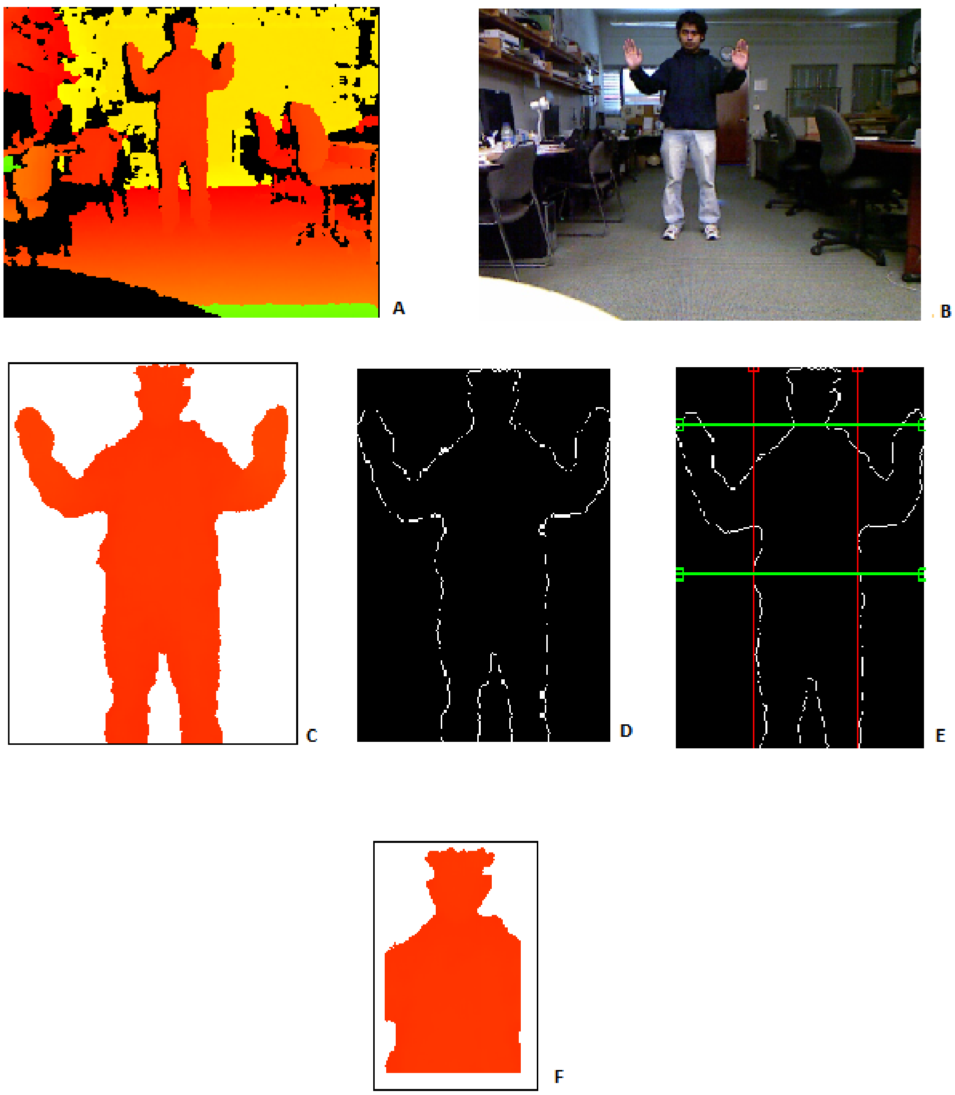


Figure 4.1: Torso Initialization from the depth and color images. Top row shows the depth image (A) and the color image (B). The the extracted foreground is shown in (C) , the edge map computed from the foreground image (D) , the result of Hough transform to detect the vertical edges (E). The resulting torso template is shown in (F).

Mean-shift Embedded Particle Filter (MSEPF) [30]. For running the mean-shift algorithm we formulate the feature-space as an augmentation of color and depth. The MSEPF algorithm is described in Section 4.6.

4.5 Skin-color Detection

There are a large number of skin detection algorithms available in the literature. They can be broadly classified into two types [12] :

i) *Frame based approach*: In frame based approaches the parameters for skin colour detection are initialised at the beginning of the algorithm and not altered thereafter. These parameters can be learned by using discriminative approach.

ii) *Sequence based approach*: In sequence based approach the parameters are adapted based on intensity values from the sequence of images. It is assumed that the skin pixels have some initial colour model. This model adapts to the changes in the illumination in the environment by using principle similar to moving average.

Based on the parameters the image pixels are assigned probability values (probability of being a skin pixel) or clustered in the space of skin colours to assign the labels. We aim to make the system illumination adaptive using a spatial prior of skin color in the given room illumination. The system is trained with the user's skin colour during the initialization of the system. Initially the locations of hands are determined using distance transform on the depth map of the outstretched arms. This process is repeated over ten frames during which hands are registered at various locations in the image. Spatial prior of skin-color for these regions are determined by computing the mean and covariance of the color of hands in these locations. Spatial map of color is computed by interpolation.

4.6 Particle Filter Implementation for Tracking Color Region

We use a particular variant of particle filter algorithm in this thesis to implement the colored blob tracking. The important component of our particle filter implementation is a local optimization step after assignment of the initial weight values to the particle. This optimization aims to move to the particles towards the local maxima. In our approach, we use the mean-shift algorithm [3] for performing this local optimization since we are using color information. The usage of mean-shift reduces the number of particles to describe a distribution [30]. Hence, it improves the time

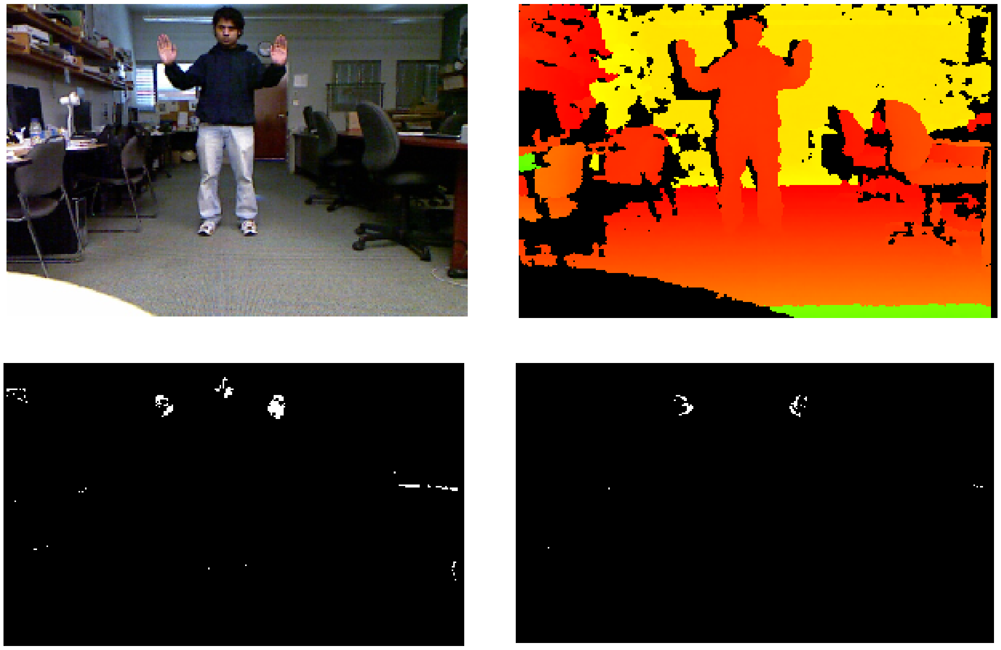


Figure 4.2: Initialization of color tracking framework using color and depth images. The top row shows the color and depth maps. The bottom row shows the skin color region (left) and the motion mask of skin colour region (right).

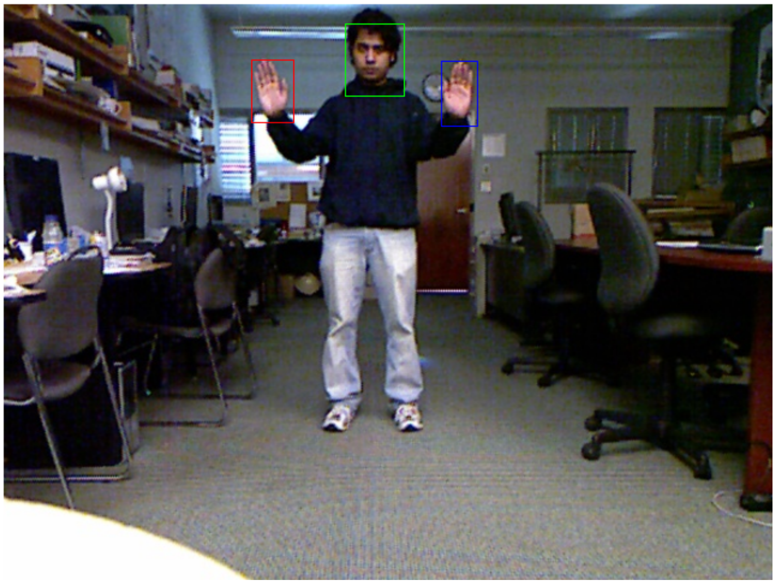


Figure 4.3: Result of color Tracking. Three bounding box shows the result of three regions of interest in the coloured image.

requirement for particle filtering. This method is known as mean-shift embedded particle filter (MSEPF). In our work, we have extended the original work from 2D tracking to 3D tracking by incorporating the range measurement from the Kinect.

The basic steps of the MSEPF algorithm are:

Step 1: Re-sampling N particles

Step 2: Propagating each particle using the Dynamic motion model

Step 3: Optimizing the particle using mean shift optimization

Step 4: Weight the particles by using a Statistical Likelihood model

Step 5: Estimate the expected position of the particles

For describing the implementation of our framework, lets say that that we want to estimate the pdf $P(\mathbf{x}_t|\mathbf{c}_{1:t}, \mathbf{d}_{1:t})$ for tracking the hand location. This uses the color information from the appearance $\mathbf{c}_{1:t}$, raw depth information $\mathbf{d}_{1:t}$. In the following subsection we give a description of our implementation for MSEPF. Equations related to mean-shift local optimization , in this subsection, are adapted from the original work [30].

4.6.1 Dynamic State Update for Particle Propagation

For tracking purpose we represent the hand by a bounding rectangle. For tracking the hand, its state at time t is described by three parameters.

$$\mathbf{x}_t = [x, y, z] \tag{4.1}$$

(x, y) represent the centroid of the current location of the hand and z represents the range measurement of the hand , required to approximate the current location of the hand in the 3D space.

We use the following motion model based on random walks [10] along X and Y axes:

$$\mathbf{x}_t = \mathbf{x}_{t-1} + \mathbf{E}_t \tag{4.2}$$

where $\mathbf{E}_t$ denotes the stochastic term associated with the zero-mean Gaussian noise. Covariance parameter associated with $\mathbf{E}_t$ is learned empirically.

4.6.2 Observation

The depth map is augmented with the color map to create the skin blob $\mathbf{M}_c$. The zero-order moment corresponding to the skin-blob is computed over all points $(\acute{x}, \acute{y})$ lying within the bounding box of the object.

$$K^c = \sum_{\acute{x}} \sum_{\acute{y}} \mathbf{M}_c(\acute{x}, \acute{y}) \tag{4.3}$$

The size of the bounding box is proportional to the depth. The constant of proportion is empirically determined after series of experiments.

However the colour cue is not the most reliable when the hand is moving quite fast. In such scenario motion cue provides a better representation of the hand. In the original paper of MSEPF [6] it was assumed that the other objects move slower than the hand. Our approach doesn't have this constraint as we isolate the foreground from the background by using the depth information. The moving region is isolated by using temporal differencing of the image. The region of interest in this case is obtained by using logical AND between the skin blob location and significant motion region to create the map $\mathbf{M}_m$. The zero-order moment is computed on this color-motion blob for all points $(\acute{x}, \acute{y})$ lying within the bounding box of the object.

$$K^m = \sum_{\acute{x}} \sum_{\acute{y}} \mathbf{M}_m(\acute{x}, \acute{y}) \tag{4.4}$$

The color and motion cues are balanced by a trade-off factor β. Thus, combining the effect of both cues one can define the zero order moment as:

$$K = (1 - \beta)K^c + \beta K^m \tag{4.5}$$

4.6.3 Mean-Shift Optimization

Given the particle locations, we compute the new location by using the Mean Shift optimization as described on the original paper [30].

Assuming that we have the initial blob centroid at (x^0, y^0), one can update the centroid location by performing the following steps:

Step 1: Compute the observed colour and motion cues over the pixels $(\acute{x}, \acute{y})$, within the rectangular bounding box located at the centroid

$$K = \sum_{\acute{x}} \sum_{\acute{y}} (1-\beta)\mathbf{M}_c(\acute{x}, \acute{y}) + \beta \mathbf{M}_m(\acute{x}, \acute{y}) \tag{4.6}$$

$$K_{\acute{x}} = \sum_{\acute{x}} \sum_{\acute{y}} \acute{x}((1-\beta)\mathbf{M}_c(\acute{x}, \acute{y}) + \beta \mathbf{M}_m(\acute{x}, \acute{y})) \tag{4.7}$$

$$K_{\acute{y}} = \sum_{\acute{x}} \sum_{\acute{y}} \acute{y}((1-\beta)\mathbf{M}_c(\acute{x}, \acute{y}) + \beta \mathbf{M}_m(\acute{x}, \acute{y})) \tag{4.8}$$

Step 2: Compute the new centroid (x^1, y^1) of the bounding box based on the zero-order and first- order moments for x and y .

$$x^1 = \frac{K_{\acute{x}}}{K}, y^1 = \frac{K_{\acute{y}}}{K} \tag{4.9}$$

Step 3: If the displacement of centroid is more than tolerance limit then go to Step 1 else terminate the loop.

4.6.4 Particle-weight Assignment

The particles are assigned weight based on probability distribution model. A similarity measure is computed based on the distance between the target region and the candidate region. This measure is used in the distribution function to obtain the probability or the weight. The dissimilarity measure $Dist$ between the i^{th}particle and the target object is estimated as follows

$$Dist(i) = (1 - \frac{K^{(i)}}{T}) \tag{4.10}$$

where T is the total number of pixels in the bounding box rectangle. Subsequently the likelihood w^i of the i^{th} particle being the representative of the target is estimated by the Gaussian distribution.

$$w^{(i)} = \frac{1}{\sigma_c\sqrt{2\pi}} e^{-(Dist(i))^2 / 2\sigma_c^2} \tag{4.11}$$

To account for the range information, the weights are adjusted as

$$w^{(i)} = w^{(i)} \exp{(-\frac{(z^{(i)} - \mathbf{R}(x^{(i)}, y^{(i)}))^2}{\sigma_d^2})} \tag{4.12}$$

where, $(x^{(i)}, y^{(i)}, z^{(i)})$ is the i^{th} particle and $\mathbf{R}$ is the range image. Thus $\mathbf{R}(x^{(i)}, y^{(i)})$ is the range data corresponding to a pixel, $(x^{(i)}, y^{(i)})$, in the color image. The weights, $w^{(i)}$, are then normalized so that their values add upto one. These weights are used to compute the expected location of the target from the generated samples.

4.7 Torso Tracking

The torso is tracked based on the template-matching approach using particle filter. We assume that the shape of the torso is invariant. The particles are initialized as the centre of gravity of the torso region. They are propagated according to first order dynamic model similar to that of MSEPF described in the earlier section. The image likelihood is computed based on a similarity measure between the shape of the template and the shape detected within the bounding box of the propagated particles. In our work we have used Sum of Square Distance (SSD) similarity between the template and the image.

$$P(\mathbf{z}_{T,t}|\mathbf{x}_{T,t}) = exp(-\frac{(SSD(\mathbf{z}_{T,t}, \mathbf{x}_{T,t}))}{\sigma_T^2}) \tag{4.13}$$

In this step we account for occlusion of the upper-body due to the movement of arms. This is done by using depth slicing (Figure 4.4). We assume that the part of the upper body, which is occluded by arms, retains the same range values from pre-occlusion stage.

4.8 Shoulder Estimation

The location of shoulder is estimated from the location of torso. The distance between the torso and shoulder is fixed and can be estimated when the torso template is initialized. At present our implementation is only equipped to handle the scenario where user is facing the camera when the tracking is underway. As a result, the planar distance between the torso key-point and shoulder remains the same at all time and is tracked by estimating the key-point corresponding to the torso.

4.9 Spatial Prior Computation

We used an example-based approach to learn the spatial prior. In our application this spatial prior is used to localize the elbow key-points when the hands can be tracked independently without any occlusion event. We learn the location of elbow with respect to corresponding hand and shoulder from manually annotated images. There are two types of challenges associated with this example-based learning. First, the number of training examples that need to be used to account for all possible configurations of the arms. Second, the prior should be invariant over different users. Instead of learning the joint configuration of all the key-points, we learn the pairwise constraints from the examples. This results in reducing the dimensionality of the problem and hence reduces its variance. We make the priors scale invariant by measuring the locations of key-points of hands and elbow relative to the head (as reference point), and representing the spatial location as unit vectors. For discretizing unit vectors in 3D we use a 20 x 20 x 20 bin. For clarity of representation, we define a mapping function $\psi_{i,j}$ for computing the vectors $\mathbf{x}_{j,t}$ when the corresponding vector $\mathbf{x}_{i,t}$ is encountered. Figure 4.5 shows the construction of prior database using exemplar image. In this illustration the joint state space of left elbow (shown in yellow) and left wrist (shown in red) is indicated. Centroid of the hand, in training examples, is determined by interpolating the wrist (in red) and the finger tips (in green).

$$\psi_{i,j}(\mathbf{x}_{i,t}; \mu_{dist}, \mathbf{x}_{H,t}) = \mu_{dist} * Table_{ij}(\widehat{\mathbf{x}_{iH,t}}) + \mathbf{x}_{H,t} \tag{4.14}$$

In this equation μ_{dist} is the mean distance between the key-points $\mathbf{x}_{i,t}$ and $\mathbf{x}_{j,t}$. $\widehat{\mathbf{x}_{iH,t}}$ represents the unit-vectors corresponding to the location of the key-point of part i with *head* of the user as the point of reference in the coordinate system.

4.10 Elbow Initialization

The elbow is initialized using the estimated hand centroid and shoulder key-points. This is done by combining the spatial prior of elbow and the observed depth map. For known hand and shoulder locations, the candidate locations of elbow are sampled using spatial prior. These samples are weighted based on image cues. Since the length of upper and lower arm are unknown during initialization, the elbow location is determined by varying the length l of upper and lower arm iteratively. This iterative process can be viewed as minimization of a cost function. This process gives us the location of elbow key-point $\mathbf{x}^*_{e,t}$ and the length of upper and lower arms, which are both initialized as l^*. This penalty function minimizes the distance between the elbow location predicted by the hand $\psi_{w,e}(\mathbf{x}_{w,t}; l, \mathbf{x}_{H,t})$ and that predicted by shoulder $\psi_{s,e}(\mathbf{x}_{s,t}; l, \mathbf{x}_{H,t}))$. Density functions $P(\mathbf{z}_{e,t}|\psi_{w,e}(\mathbf{x}_{w,t}; l, \mathbf{x}_{H,t}))$ and $P(\mathbf{z}_{e,t}|\psi_{s,e}(\mathbf{x}_{s,t}; l, \mathbf{x}_{H,t}))$ are used to incorporate the range data. The penalty function is given by

$$\begin{aligned}(l^*, \mathbf{x}^*_{e,t}) = arg\, min_l\{(\psi_{w,e}(\mathbf{x}_{w,t}; l, \mathbf{x}_{H,t}) - \psi_{s,e}(\mathbf{x}_{s,t}; l, \mathbf{x}_{H,t}))\\ P(\mathbf{z}_{e,t}|\psi_{w,e}(\mathbf{x}_{w,t}; l, \mathbf{x}_{H,t}))P(\mathbf{z}_{e,t}|\psi_{s,e}(\mathbf{x}_{s,t}; l, \mathbf{x}_{H,t}))\}\end{aligned} \quad (4.15)$$

4.11 Elbow Estimation and Tracking

We combine the elbow dynamic motion model with the image information to obtain the Importance sampling function for generating the particles to track the elbows. We take advantage of the extracted foreground depth map and use the length of upper and lower arm to get image observation of elbow with respect to the corresponding shoulder and hand. We construct a mixture density function by combining the dynamic model and range features as following :

$$Imp(\mathbf{x}_{e,t}) = P(\mathbf{x}_{e,t}|\mathbf{x}_{e,t-1})\psi_{w,e}(\mathbf{x}_{w,t}; ll, \mathbf{x}_{H,t})\psi_{s,e}(\mathbf{x}_{s,t}; ul, \mathbf{x}_{H,t}) \quad (4.16)$$

where ll and ul denotes the length of lower and upper arm respectively as computed from previous frame.

When the hands and face occlude one another, it is difficult to estimate the location of centroid of hand. Consequently, the hand location is not used to localize the elbow. For this reason, under the occlusion condition, the importance sampling function is modified as the Gaussian density corresponding to the lower arm is dropped from the importance sampling function.

$$Imp_{occl}(\mathbf{x}_{e,t}) = P(\mathbf{x}_{e,t}|\mathbf{x}_{e,t-1})\psi_{s,e}(\mathbf{x}_{s,t}; ul, \mathbf{x}_{H,t}) \tag{4.17}$$

For computing the weights, we use the weighted distance transform map computed from non-Torso foreground pixels. We define the observation likelihood function $P(\mathbf{z}_{e,t}|\mathbf{x}_{e,t})$ as :

$$P(\mathbf{z}_{e,t}|\mathbf{x}_{e,t}) = exp(-(Euc(\mathbf{z}_{e,t}, \mathbf{x}_{e,t}))) \tag{4.18}$$

The density function $P(\mathbf{x}_{j,t}|\mathbf{x}_{e,t})$ is computed as a normal distribution , where variance $\sigma_{e,j}$ is learned from training examples, as in :

$$P(\mathbf{x}_{j,t}|\mathbf{x}_{e,t}) = \mathcal{N}(Euc(\mathbf{x}_{j,t}, \psi_{e,j}); 0, \sigma_{e,j}) \tag{4.19}$$

where, $\sigma_{e,j}$ is the allowed variance in distance. This is determined empirically. Figure 4.6 shows an illustration of Elbow tracking. Top row shows the result for tracking without occlusion. The importance sampling function used the left hand and left shoulder to generate the elbow samples. For tracking under occlusion, the left hand could no longer be used to generate samples, as a result the particles spread out considerably as seen in the second row of the figure.

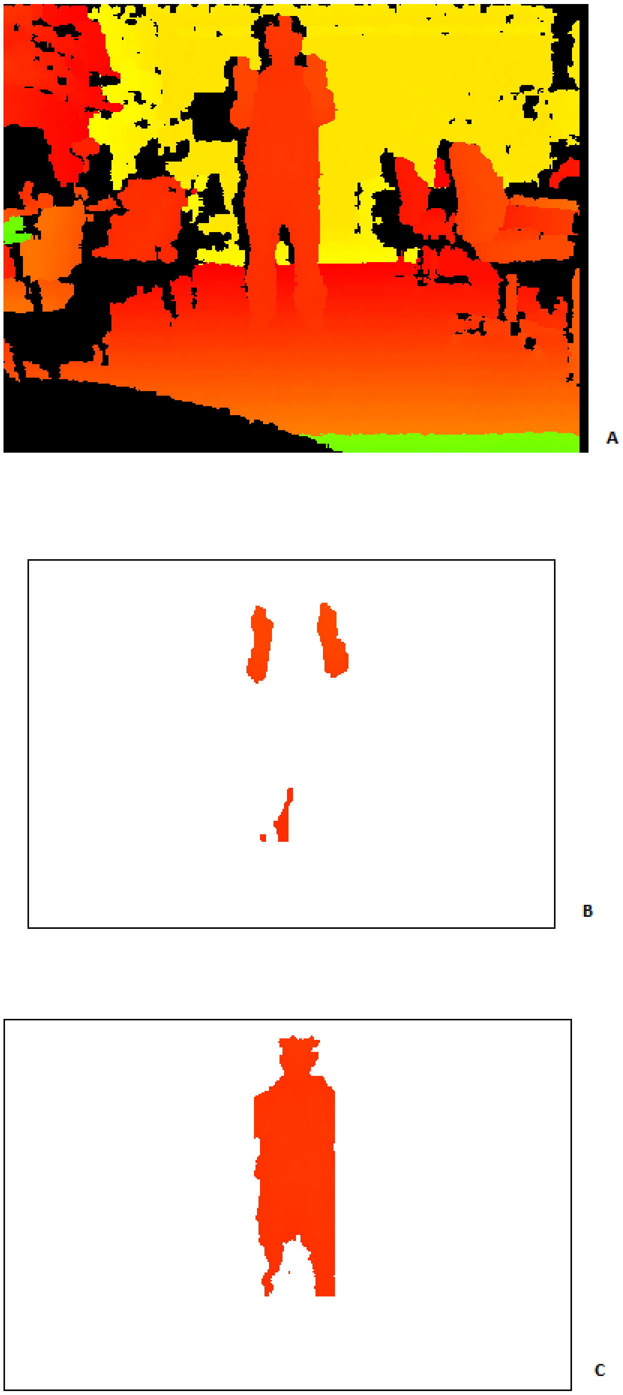


Figure 4.4: Illustration of depth-slicing for torso recovery. (A) shows the depth image where the arms occlude the torso, The arm region is sliced out by using relative depth of torso and hand (B), the recovered map of torso (C).

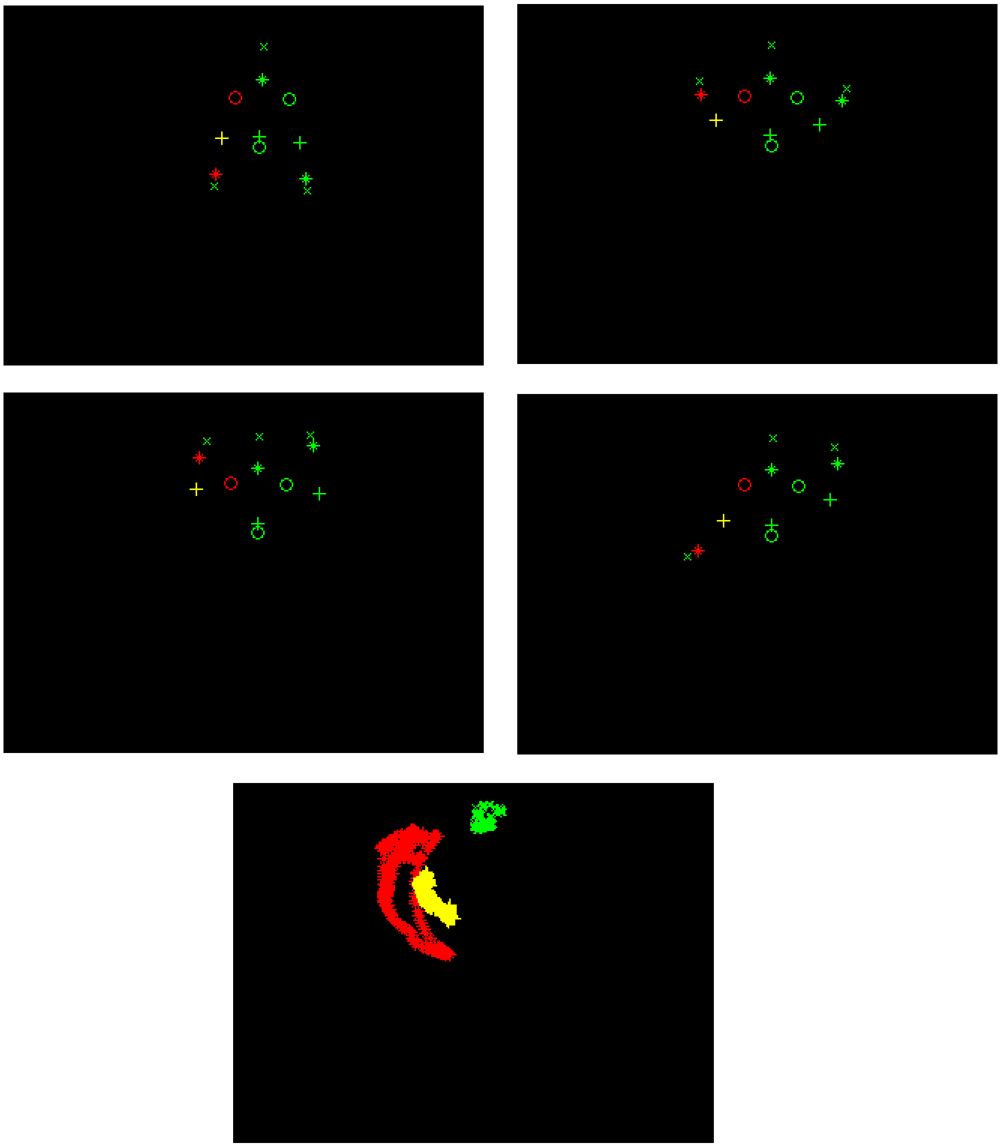


Figure 4.5: Construction of joint prior. The location of left elbow is learned with respect to user's left hand. User's face is used as point of reference to compute the unit vectors, so that normalized data can be referenced from the table.

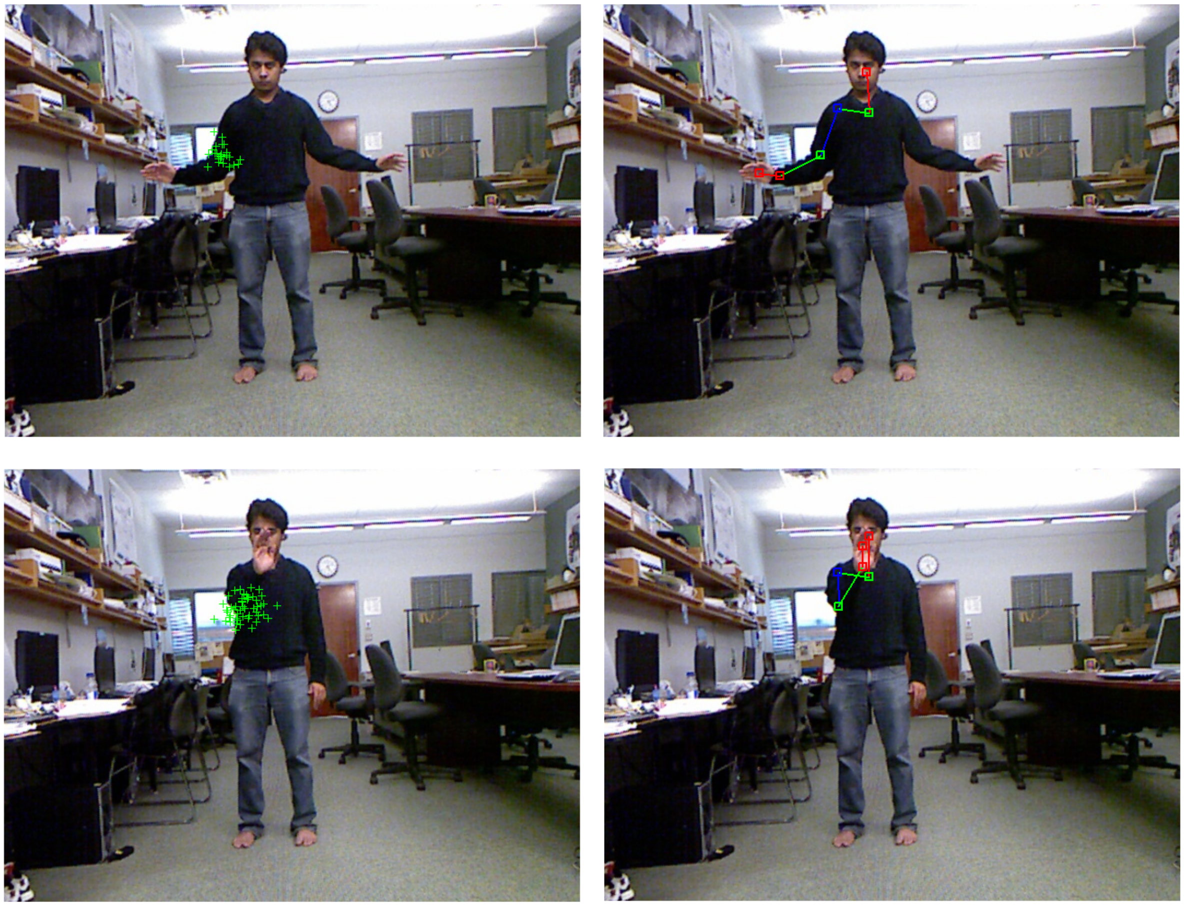


Figure 4.6: Illustration of elbow tracking without occlusion (first row) and with occlusion (second row). Note the increase in variance when the knowledge about hand location is discarded during occlusion event.

Chapter 5

Experimental Results

The proposed system has been implemented in Matlab R2010 on a Core 2 Duo Processor (T60 2.10GHz). Thus far the results have been tested on datasets of one user. The range and color images were collected using Microsoft Kinect. The results of the tracker are discussed subsequently.

We collected 2 datasets, each with 120 frames. In the first dataset, we consider the scenario when only one hand creates the occlusion event when it comes in front of the user's face. In the second video both the right and left hand generate the occlusion event when they interact with one another and with the face.

Figure 5.1 shows the result of color tracking in absence of occlusion event. For each tracker 25 particles have been used. Figure 5.2 shows the result of head and torso tracking in the corresponding frames. The results of torso tracking, shown in the images, are obtained by using the SSD based shape matching algorithm. When the torso gets occluded by hand, depth slicing strategy was used, as we discussed in previous chapter.

Finally in Figure 5.3, we show the result of elbow tracking based on the independent tracking results obtained from the 4 trackers (3 color-tracker and 1 shape-tracker). The results reflect the dependency of elbow with its neighbouring parts, hand and shoulder.

In our first experiment we compared the result of our algorithm, while tracking two objects (left hand and head), with multiple independent trackers using Mean Shift embedded particle filter [30], Markov Chain Monte Carlo Tracker [13], Joint Particle Filter [8] , multiple independent particle filters [10]. For this comparison

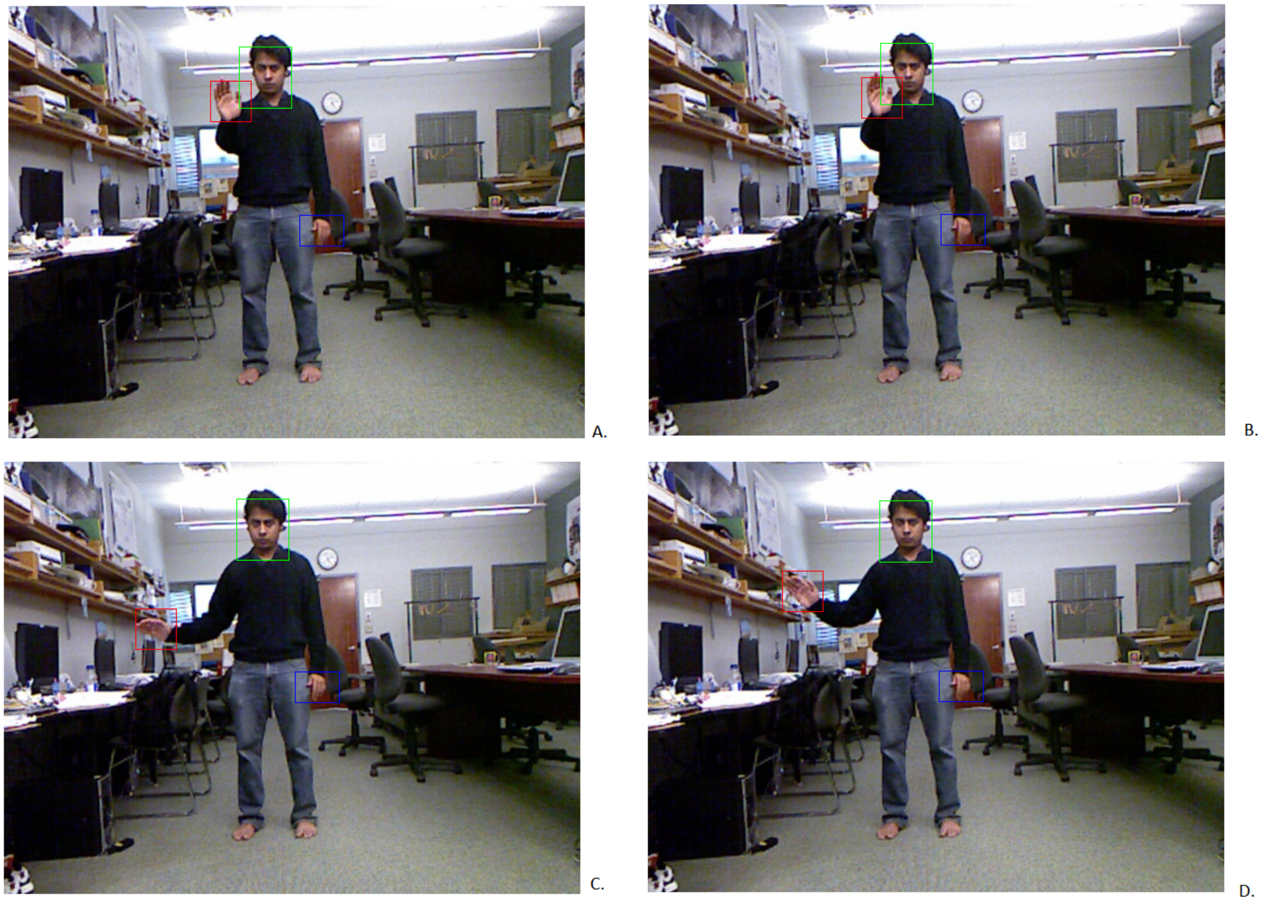


Figure 5.1: Results of MSEPF trackers for skin coloured regions. The three regions have been shown using three different colours. The state of the trackers are represented by the center of the respective bounding boxes.

we considered the first image sequence where the left hand occludes the user's face. We show the result of our algorithm under occlusion. Figures 5.4 and 5.5 show the comparison of the trackers while tracking the head and left hand respectively during the occlusion event. Both the plots show an improvement of tracking performance over depth-based mean shift embedded particle filter algorithm. We estimated the error, in terms of distance between estimated object centroids and manually annotated object centroid, over 70 frames and show the plots here. The performance of our tracking algorithm can be visualized in Figure 5.6 and 5.7. We show the visual comparison of our result with respect to MSEPF. In Figure 5.6, we compare the

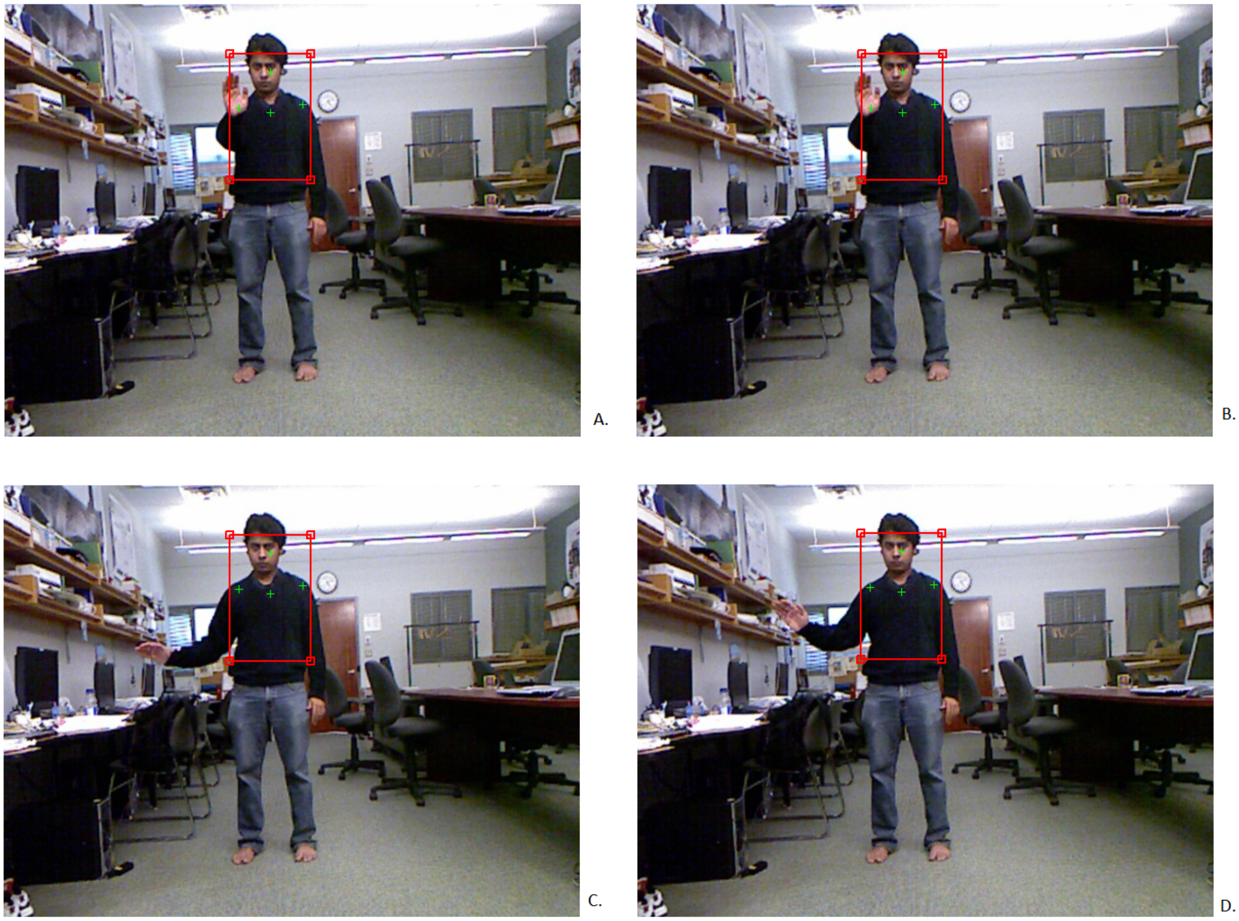


Figure 5.2: Results of torso tracking using SSD based template matching and depth-slicing. The three cross-marks represent 3 landmarks of the body. The left-most cross represents the left shoulder, the right-most cross represents the right-shoulder and central cross represents the state of the torso.

results of the algorithms when the occlusion event is beginning. Figure 5.7 shows the performance when the occlusion event terminates. For both the scenarios our algorithm shows better performance while avoiding tracker fusion.

Table 5.1 shows the quantitative comparison of the algorithms using Root Mean Squared Error (RMSE) metric. RMSE is taken as an average over 5 runs on each image sequence. One can see that our algorithm performs significantly better than the baseline algorithms (multiple independent tracker, based on particle filters, with or without mean shift local optimization) and comparable to the performance of

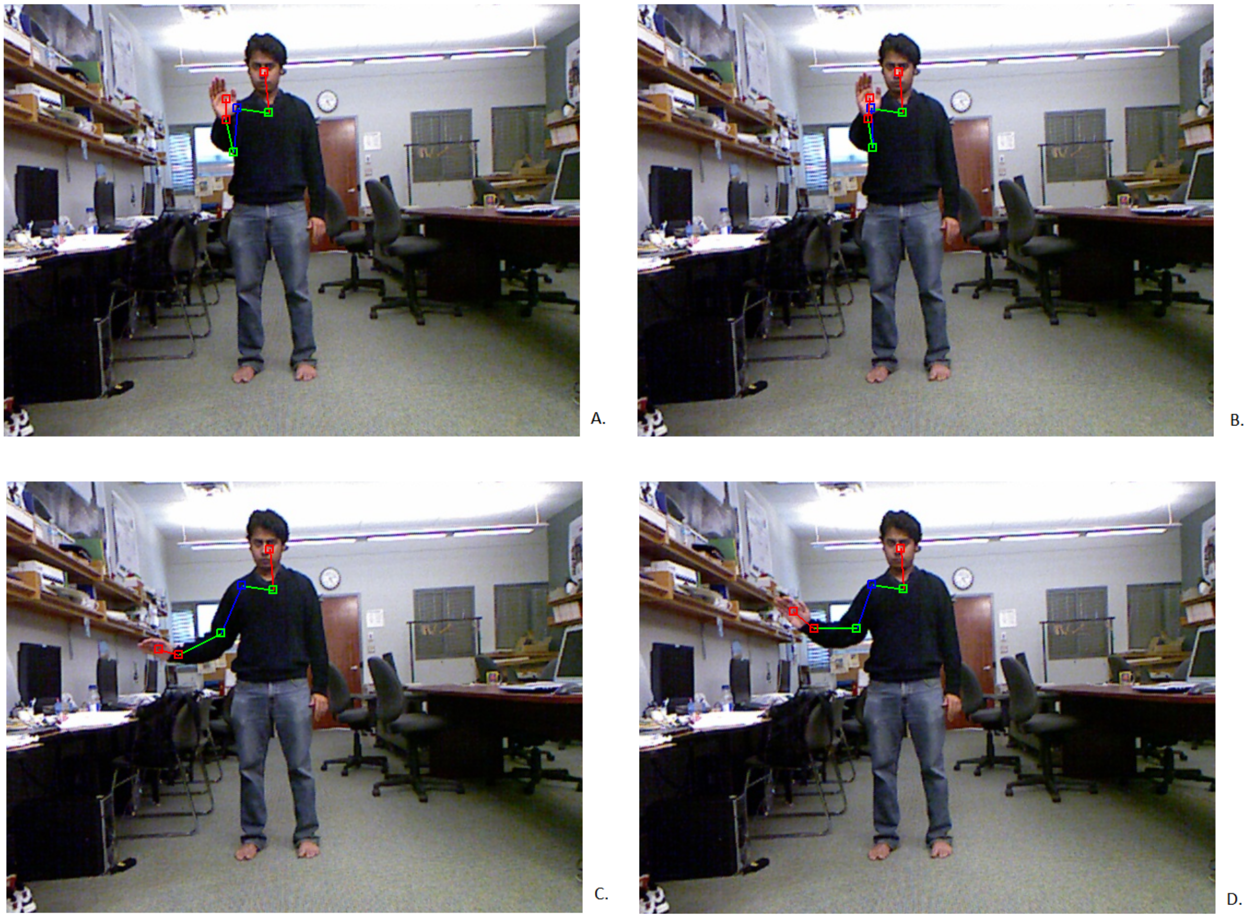


Figure 5.3: Result of estimation of elbow from the estimated states of hand and torso. The joints locations are joined and overlapped on the color image to show the correspondences visually.

MCMC multi-object tracking algorithm.

In the second experiment we use our algorithm to track both hands and face simultaneously. For this experiment 150 particles are used during occlusion reasoning and 4 iterations of particle updating were used. Figure 5.8 shows how the particles converge with each iteration. The results indicate that our algorithm successfully track the hands during occlusion even when they are close to the face. We show the results over 8 consecutive frames in Figure 5.9. We plot the tracking error while tracking the left hand, right hand and head (shown in Figure 5.10, 5.11 and 5.12) and compare the results with multiple independent trackers (with and without mean-shift). The RMS errors, generated by the algorithms, are listed in Table 5.2. RMS error for Independent trackers of hand and right hand are high because of tracker fusion during occlusion event.

In our third experiment we demonstrate that our algorithm can perform tracking under poor illumination (see Figure 5.13). In this phase all the trackers relies exclusively on depth information as skin colour information is not available. Under this condition we had to impose two restrictions on our framework. First, the face must be visible in the color camera so that face detection can be performed to initialize trackers. Second, the hands cannot stay close the body to ensure that depth slicing and torso subtraction don't remove any information related to hand and elbow. We avoid using skin-color information for the corresponding tracker when the skin color detector fails to detect the color of the corresponding object.

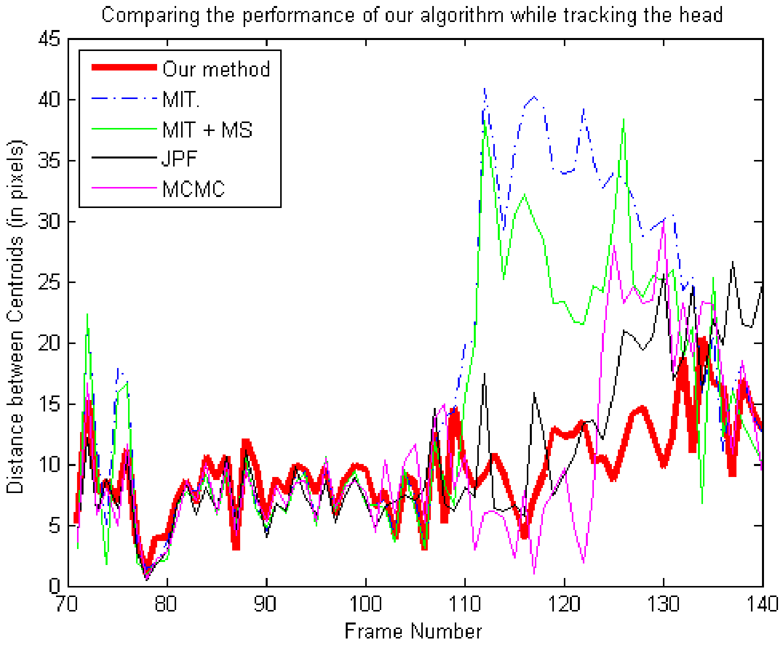


Figure 5.4: Plot showing the comparison between the tracking error of head location generated by our algorithm in first experiment. Error is computed as the distance between estimated location of the object centroid and the manually annotated ground truth.

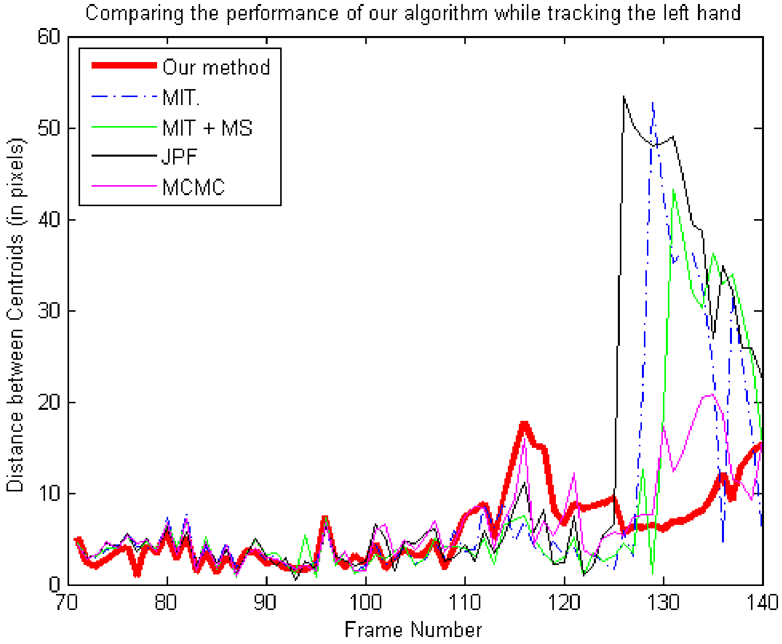


Figure 5.5: Plot showing the comparison between the Tracking Errors of Left Hand location generated by our algorithm in first experiment. Error is computed as the distance between estimated location of the object centroid and the manually annotated ground truth.

Table 5.1: Comparing RMS Error of the algorithms for Experiment 1

Algorithms	Left Hand	Head
M.I.T.	8.3	17.15
M.I.T + meanshift	7.99	14.41
JPF	11.60	10.83
MCMC	6.55	10.38
Our algorithm	5.86	9.6

Table 5.2: Comparing RMS Error of the algorithms for Experiment 2

Algorithms	Left Hand	Head	Right Hand
M.I.T.	10.39	24.92	23.01
M.I.T + meanshift	10.29	17.42	22.95
MCMC	27.05	9.43	5.73
Our Algorithm	10.98	7.18	8.55

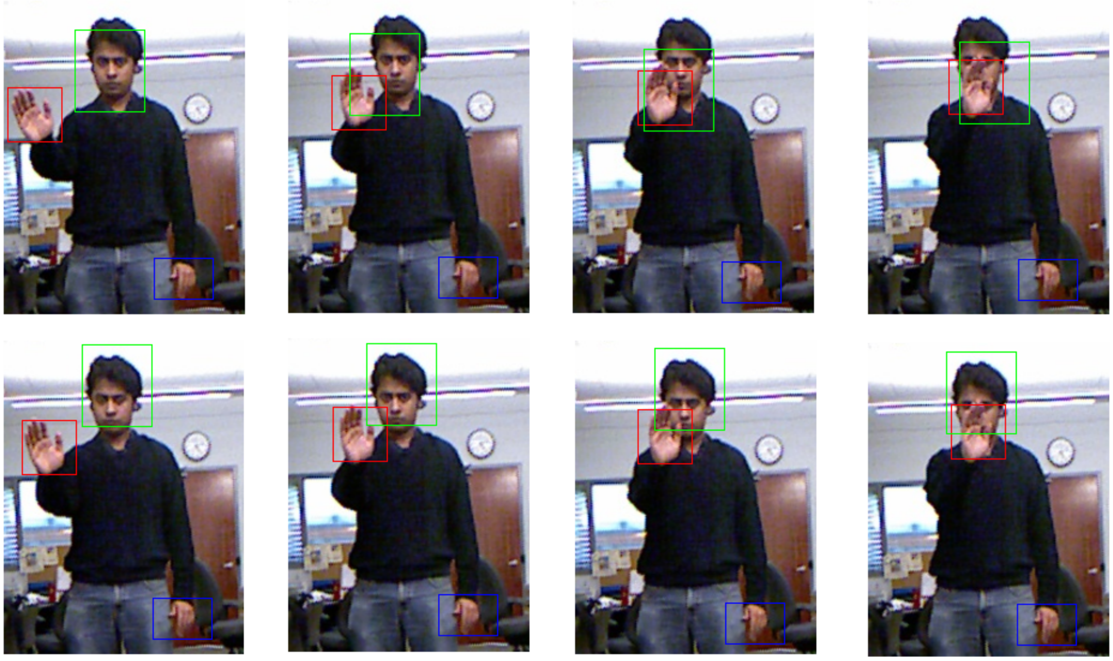


Figure 5.6: Comparison of the tracking performances when the occlusion event initiates. Top row shows the result of independent trackers. Bottom row shows the result of our algorithm.

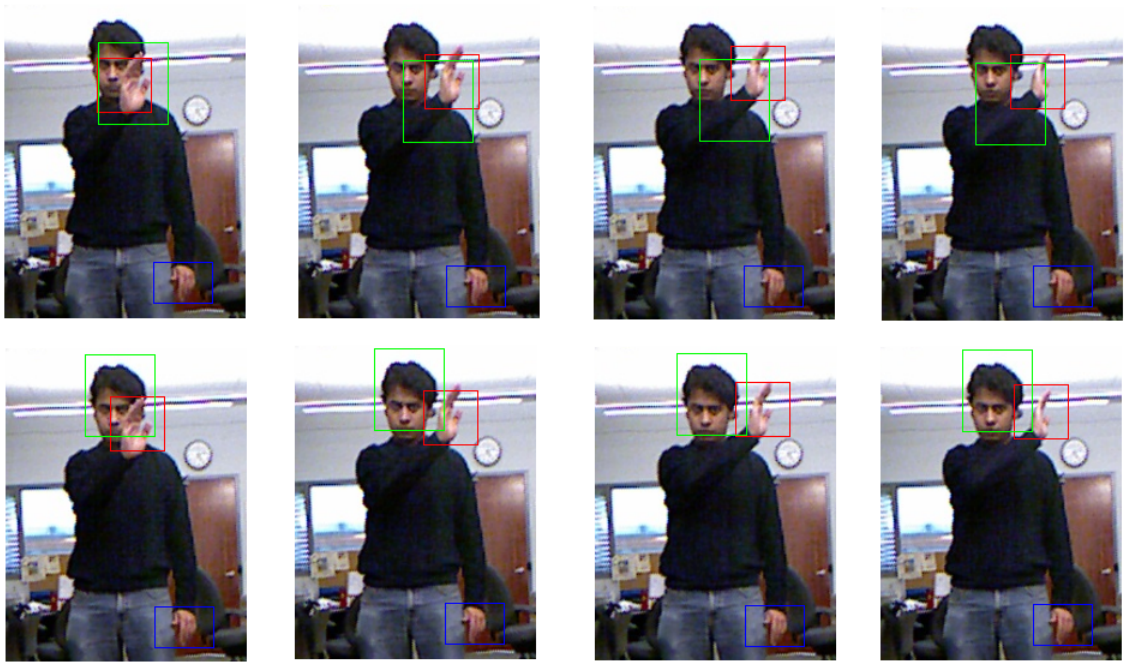
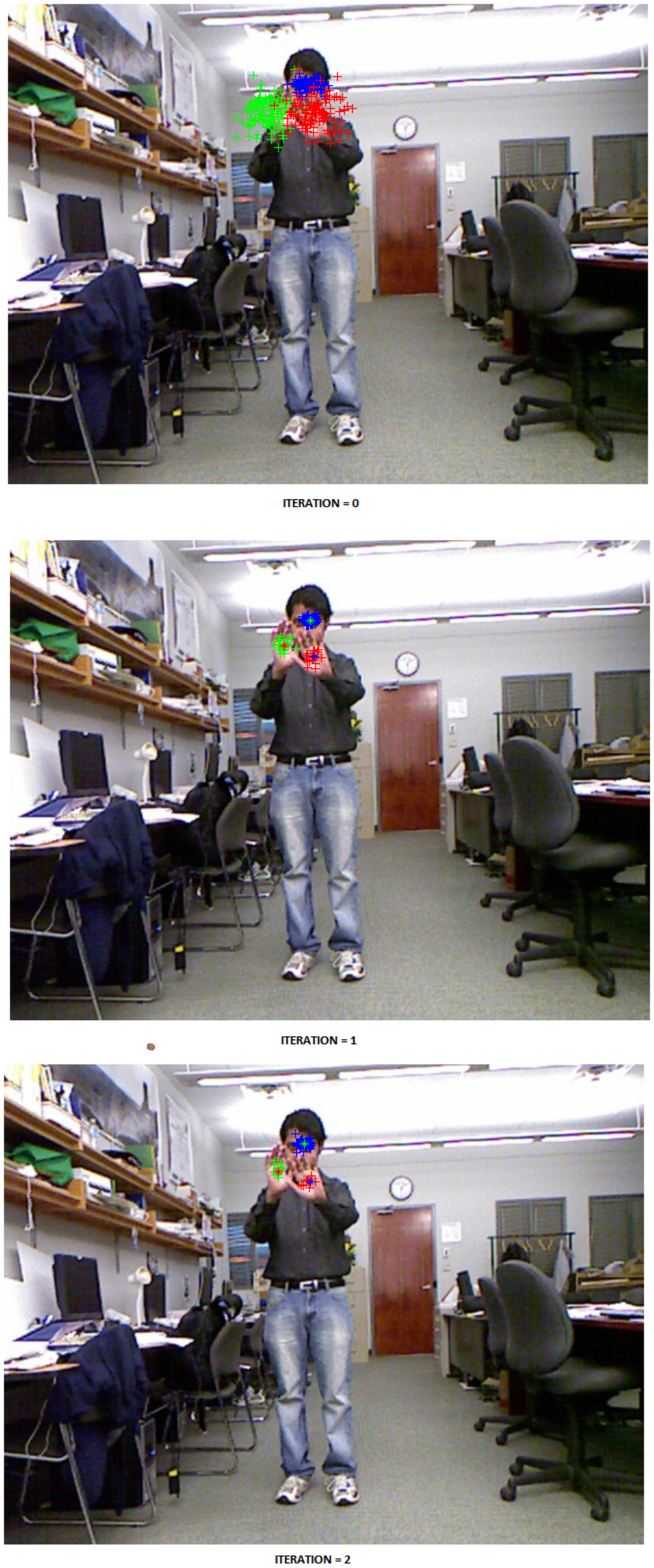


Figure 5.7: Comparison of the tracking performances when the occlusion event terminates. Top row shows the result of independent trackers. Bottom row shows the result of our algorithm.



ITERATION = 0

ITERATION = 1

ITERATION = 2

Figure 5.8: Convergence of particles after every iteration.

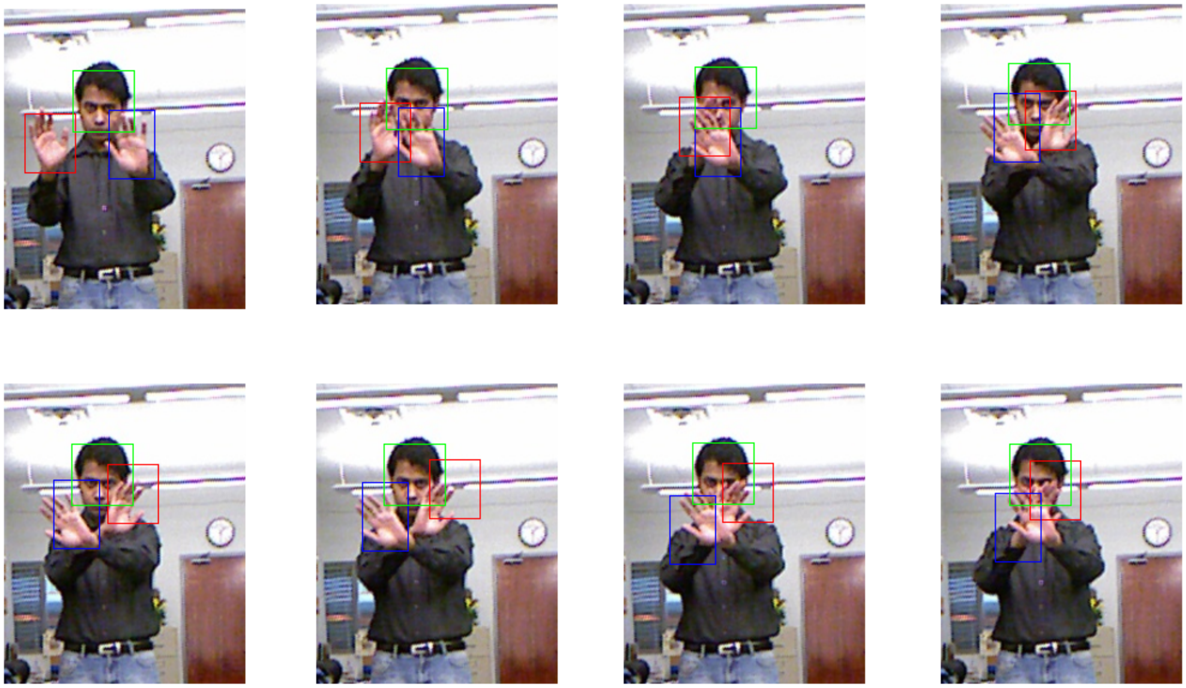


Figure 5.9: Results of our algorithm while tracking three interactive targets (left hand, right hand and face) simultaneously.

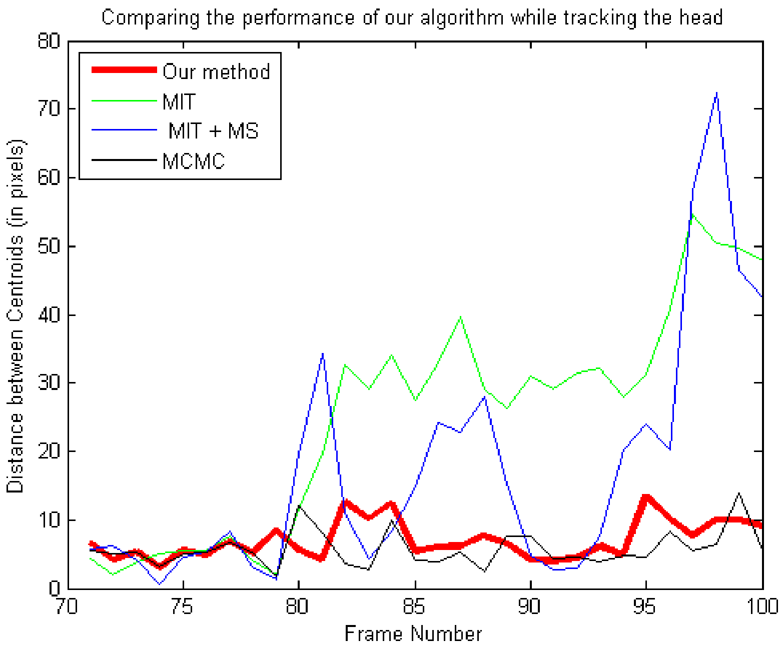


Figure 5.10: Plot showing the comparison between the tracking error of head location generated by our algorithm and the multiple independent trackers (M.I.T.) during the second experiment. Error is computed as the distance between estimated location of the object centroid and the manually annotated ground truth.

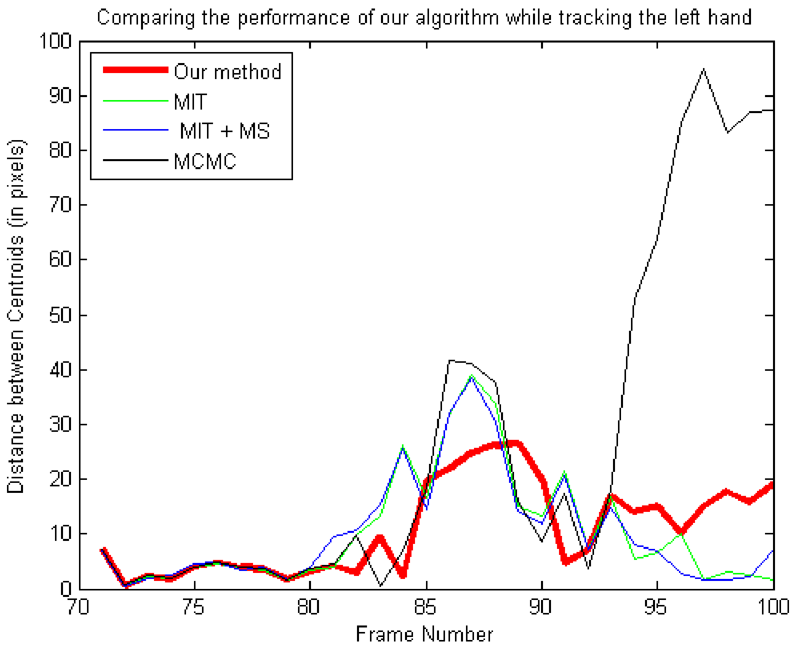


Figure 5.11: Plot showing the comparison between the tracking errors of left hand location generated by our algorithm and multiple independent trackers during the second experiment. Error is computed as the distance between estimated location of the object centroid and the manually annotated ground truth.

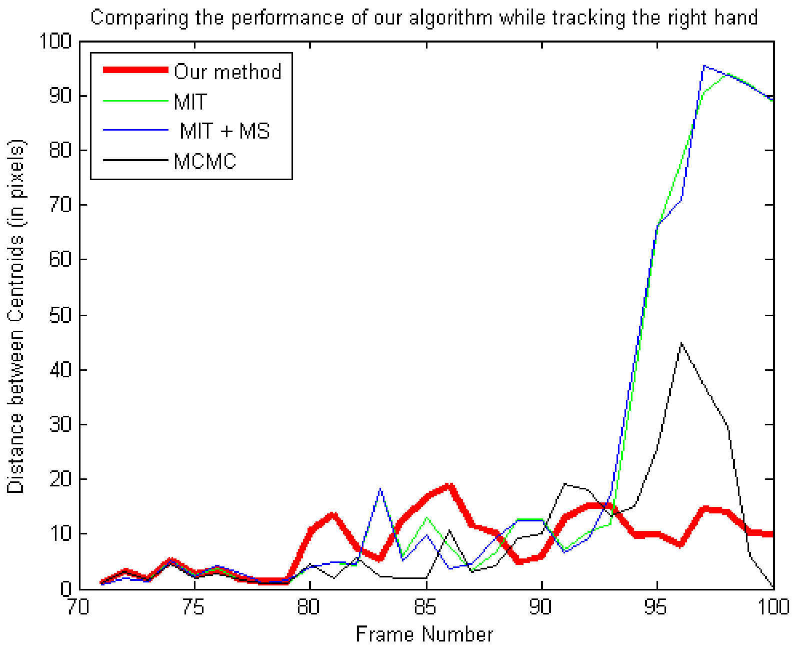


Figure 5.12: Plot showing the comparison between the tracking errors of right hand location generated by our algorithm and multiple independent trackers during the second experiment. Error is computed as the distance between estimated location of the object centroid and the manually annotated ground truth.

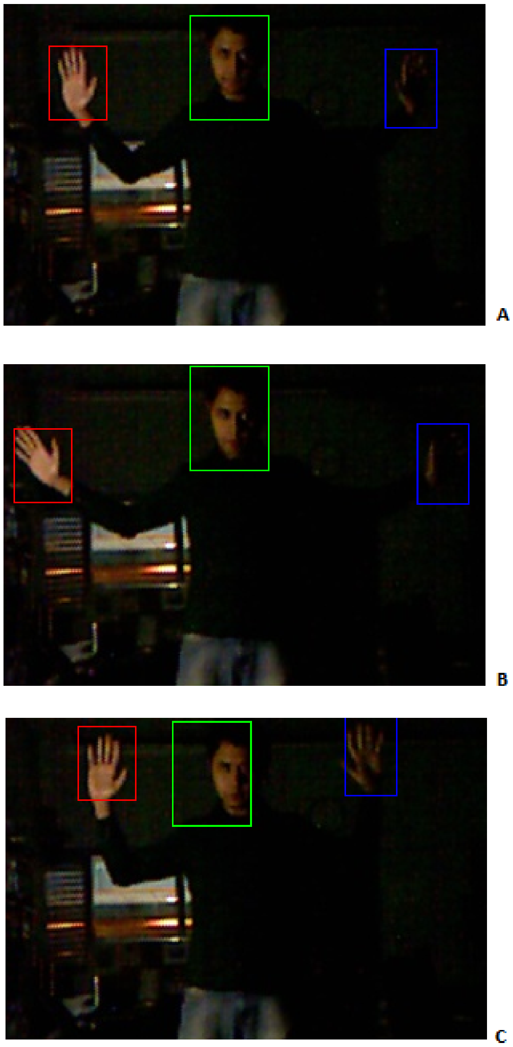


Figure 5.13: Illustration of tracking under low illumination.

Chapter 6

Conclusion and Future Work

Hand tracking has a wide variety of applications because of increasing use of gesture recognition systems for practical applications. While there has been substantial research in hand tracking in video, the advent of real time range sensors has spurred on further development. In this work we have proposed a theoretical framework for hand tracking by fusing the data from a range sensor and a color camera.

In this thesis we have developed a multi-object tracking algorithm using Dynamic Bayesian Network for tracking hands and face. The main contribution of this thesis is the integration of the appearance based color trackers within the pose estimation framework. The primary implication of such approach being, color regions can be tracked independent of one another, when they are not interacting. However, when they get close to each other, additional pose information from the body can be used to improve the performance of the tracker.

Our algorithm shows improvements over skin color based hand trackers, even after incorporating depth information in these algorithms. This is because, we account for the pose information of the user within our computational framework. Experimental results indicate that our tracker performs consistently even when the hands are close to another, as well as occluding the user's face. Our algorithm is an improvement over trackers which uses depth map but not colour image. This is because trackers which use depth map alone are either computationally expensive due to pixel classification prior to tracking, or inaccurate when the hands moves close to torso and cannot be differentiated from the depth-map.

Another contribution is the development of an inter-tracker interaction density for multi-target tracking. We have exploited the availability of range sensor data to improve the likelihood measure by introducing a depth ordering factor. This additional factor rewards the hypotheses pairs based on their relative depth difference, in spite the object being close to one another.

There are some areas of improvement in our current framework. Currently the implementation has been done in MATLAB. It will be interesting to see the speed-up, when the framework is implemented in GPU to exploit the decentralized nature of the inference methodology. Also in our current implementation the user is user is assumed to stand still during the tracking process and the upper-body is assumed to be upright. This can be improved by adding additional parameters to describe the latent variables in our graphical model. We are also currently studying the change in quality of tracking as the user moves closer to or further away from Kinect sensor.

One of the areas of improvements is the importance sampling function for generating the particles for dependent trackers. In this work, we have combined the knowledge of arm length with the dynamic motion model for creating the importance sampling functions. Other image cues (edge , shape for example) can be introduced in the importance sampler to improve the quality of samples.

Feature selection is another relatively open area that one could focus on to improve the tracking performance. In our work, we have used the colour feature because of its shape and scale invariance. However, this makes it difficult to differentiate between hands when they are very close.

Our system, as of now, works with the user facing the camera front face and is used to track the hand and face region. The color tracking based framework is not yet able to handle the general scenario when the skin of the forearms is visible. A better likelihood function can be designed to make our system more robust in more complex environmental conditions.

Bibliography

[1] Ankur Agarwal and Bill Triggs. 3d human pose from silhouettes by relevance vector regression. In *In CVPR*, pages 882–888, 2004.

[2] P. Buehler, M. Everingham, D. P. Huttenlocher, and A. Zisserman. Long term arm and hand tracking for continuous sign language TV broadcasts. In *Proceedings of the British Machine Vision Conference*, 2008.

[3] Dorin Comaniciu, Peter Meer, and Senior Member. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:603–619, 2002.

[4] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Pictorial structures for object recognition. *Int. J. Comput. Vision*, 61:55–79, January 2005.

[5] Varun Ganapathi, Christian Plagemann, Sebastian Thrun, and Daphne Koller. Real time motion capture using a single time-of-flight camera. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, San Francisco, CA, USA, June 2010.

[6] Daniel Grest, Volker Krger, and Reinhard Koch. Single view motion tracking by depth and silhouette information. In *In Scandinavian Conference on Image Analysis (SCIA07)*, 2007.

[7] Gang Hua, Ming-Hsuan Yang, and Ying Wu. Learning to estimate human pose with data driven belief propagation. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 2:747–754, 2005.

[8] M. Isard. Pampas: real-valued graphical models for computer vision. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 1, pages I–613 – I–620 vol.1, june 2003.

[9] M. Isard and J. MacCormick. Bramble: A bayesian multiple-blob tracker. *Computer Vision, IEEE International Conference on*, 2:34, 2001.

[10] Michael Isard and Andrew Blake. Condensation - conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29(1):5–28, 1998.

[11] Shanon X. Ju, Michael J. Black, and Yaser Yacoob. Cardboard people: A parameterized model of articulated image motion. *Automatic Face and Gesture Recognition, IEEE International Conference on*, 0:38, 1996.

[12] P. Kakumanu, S. Makrogiannis, and N. Bourbakis. A survey of skin-color modeling and detection methods. *Pattern Recogn.*, 40:1106–1122, March 2007.

[13] Zia Khan, Tucker R. Balch, and Frank Dellaert. Mcmc-based particle filtering for tracking a variable number of interacting targets. *IEEE Trans. Pattern Anal. Mach. Intell.*, 27(11):1805–1918, 2005.

[14] Steffen Knoop, Stefan Vacek, and Rüdiger Dillmann. Sensor fusion for 3d human body tracking with an articulated 3d body model. In *ICRA*, pages 1686–1691, 2006.

[15] K. Kwon, H. Zhang, and F. Dornaika. Hand pose recovery with a single video camera. In *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, volume 2, pages 1194 – 1200 vol.2, 2001.

[16] A. Micilotta, E. Ong, and R. Bowden. Detection and tracking of humans by probabilistic body part assembly. Technical report, University of Surrey, september 2005.

[17] Thomas B. Moeslund, Adrian Hilton, and Volker Krüger. A survey of advances in vision-based human motion capture and analysis. *Comput. Vis. Image Underst.*, 104:90–126, November 2006.

[18] G. Mori and J. Malik. Estimating human body configurations using shape context matching. In *European Conference on Computer Vision LNCS 2352*, volume 3, pages 666–680, 2002.

[19] Pan Pan and Dan Schonfeld. Video tracking based on sequential particle filtering on graphs. *IEEE Transactions on Image Processing*, 20(6):1641–1651, 2011.

[20] Vladimir I. Pavlovic, Rajeev Sharma, and Thomas S. Huang. Visual interpretation of hand gestures for human-computer interaction: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19:677–695, 1997.

[21] Wei Qu and Dan Schonfeld. Real-time decentralized articulated motion analysis and object tracking from videos. *IEEE Transactions on Image Processing*, 16(8):2129–2138, 2007.

[22] Wei Qu, Dan Schonfeld, and Magdi A. Mohamed. Real-time distributed multi-object tracking using multiple interactive trackers and a magnetic-inertia potential model. *IEEE Transactions on Multimedia*, 9(3):511–519, 2007.

[23] Deva Ramanan and D. A. Forsyth. Finding and tracking people from the bottom up. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference*, 2:467, 2003.

[24] Deva Ramanan and Cristian Sminchisescu. Training deformable models for localization. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference*, 1:206–213, 2006.

[25] James Rehg and Takeo Knade. Visual tracking of high dof articulated structures: an application to human hand tracking. In *In European Conference on Computer Vision*, pages 35–46. Springer-Verlag, 1994.

[26] James M. Rehg and Takeo Kanade. Model-based tracking of self-occluding articulated objects. In *ICCV*, pages 612–617, 1995.

[27] Xiaofeng Ren, Alexander C. Berg, and Jitendra Malik. Recovering human body configurations using pairwise constraints between parts. *Computer Vision, IEEE International Conference on*, 1:824–831, 2005.

[28] Loren Arthur Schwarz, Artashes Mkhitaryan, Diana Mateus, and Nassir Navab. Estimating human 3d pose from time-of-flight images based on geodesic distances and optical flow. In *FG*, pages 700–706, 2011.

[29] Gregory Shakhnarovich, Paul Viola, and Trevor Darrell. Fast pose estimation with parameter-sensitive hashing. In *Proceedings of the Ninth IEEE International Conference on Computer Vision - Volume 2*, ICCV '03, pages 750–, 2003.

[30] Caifeng Shan, Yucheng Wei, Tieniu Tan, and Frdric Ojardias. Real time hand tracking by combining particle filtering and mean shift. In *FGR'04*, pages 669–674, 2004.

[31] Chunhua Shen, Anton van den Hengel, Anthony Dick, and Michael J. Brooks. 2D articulated tracking with dynamic Bayesian networks. In *4th International Conference on Computer and Information Technology (CIT'04)*, pages 130–136, Wuhan, China, September 2004.

[32] Jamie Shotton and Toby Sharp. Real-time human pose recognition in parts from single depth images. *Training*, 2:1297–1304, 2011.

[33] Matheen Siddiqui. Human pose estimation from a single view point , real-time range sensor. *Image Rochester NY*, (June):1–8, 2010.

[34] Hedvig Sidenbladh, Michael J. Black, and David J. Fleet. Stochastic tracking of 3d human figures using 2d image motion. In *Proceedings of the 6th European Conference on Computer Vision-Part II*, ECCV '00, pages 702–718, 2000.

[35] Leonid Sigal, Michael Isard, Benjamin H. Sigelman, and Michael J. Black. Attractive people: Assembling loose-limbed models using non-parametric belief propagation. In *NIPS*, 2003.

[36] B. Stenger, P. R. S. Mendoni, and R. Cipolla. Model-based 3d tracking of an articulated hand. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference*, 2:310, 2001.

[37] Erik B. Sudderth, Alexander T. Ihler, William T. Freeman, and Alan S. Willsky. Nonparametric belief propagation. In *CVPR (1)*, pages 605–612, 2003.

[38] Junqiu Wang and Yasushi Yagi. Adaptive mean-shift tracking with auxiliary particles. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 39(6):1578–1589, 2009.

[39] Ying Wu, Gang Hua, and Ting Yu. Tracking articulated body by dynamic markov network. *Computer Vision, IEEE International Conference on*, 2:1094, 2003.

[40] Ying Wu, John Y. Lin, and Thomas S. Huang. Capturing natural hand articulation. In *ICCV*, 2001.

[41] Ting Yu and Ying Wu. Decentralized multiple target tracking using netted collaborative autonomous trackers. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 1:939–946, 2005.

[42] Youding Zhu and Kikuo Fujimura. Constrained optimization for human pose estimation from depth sequences. In *ACCV (1)*, pages 408–418, 2007.

[43] Youding Zhu and Kikuo Fujimura. Bayesian 3d human body pose tracking from depth image sequences. In *ACCV (2)*, pages 267–278, 2009.