Journal of Real-Time Image Processing manuscript No.
(will be inserted by the editor)

Hairong Jiang · Juan P. Wachs · Bradley S. Duerstock

Real-time facilitated hands gesture recognition based interfaces for individuals with upper-level spinal cord injuries

Received: date / Revised: date

Abstract This paper presents a hand gesture based interface to facilitate interaction with individuals with upper-level spinal cord injuries, and offers an alternative way to perform "hands-on" laboratory tasks. The presented system consists of four modules: hand detection, tracking, trajectory recognition, and actuated device control. A 3D particle filter framework based on color and depth information is proposed to provide a more efficient solution to the independent face and hands tracking problem. More specifically, an interaction model utilizing spatial and motion information was integrated into the particle filter framework to tackle the "false merge" and "false labeling" problem through hand interaction and occlusion. To obtain an optimal parameter set for the interaction model, a neighborhood search algorithm was employed. An accuracy of 98.81% was achieved by applying the optimal parameter set to the tracking module of the system. Once the hands were tracked successfully, the acquired gesture trajectories were compared with motion models. The dynamic time warping (DTW) method was used for signals' time alignment, and they were classified by a CONDENSATION algorithm with a recognition accuracy of 97.5%. In a validation experiment, the decoded gestures were passed as commands to a mobile service robot and a robotic arm to perform simulated laboratory tasks. Control policies using the gestural control were studied and optimal policies were selected to achieve optimal performance. The computational cost of each system module demonstrated a real-time performance.
Keywords: Gesture recognition; 3D particle filter; Neighborhood search; dynamic time warping (DTW); CONDENSATION.

This work was partially funded by the National Institutes of Health NIH Director's Pathfinder Award to Promote Diversity in the Scientific Workforce, grant number DP4-GM096842-01.

Hairong Jiang
School of Industrial Engineering, Purdue University, West Lafayette, IN 47907, USA
E-mail: jiang115@purdue.edu

Juan P. Wachs
School of Industrial Engineering, Purdue University, West Lafayette, IN 47907, USA
E-mail: jpwachs@purdue.edu

Bradley S. Duerstock
School of Industrial Engineering and Weldon School of Biomedical Engineering, Purdue University, West Lafayette, IN 47907, USA
E-mail: bsd@purdue.edu

1 Introduction

Voice, facial expressions, gaze and hand gestures have been widely employed as communication channels for human computer interaction (HCI) and human robot interaction (HRI) [1]. These modalities of interaction have gradually made their way into assistive technologies (AT) domain, such as home medical alert systems (use abnormal behavior recognition) and intelligent wheelchairs (use gesture control). Such applications are designed to help people with disabilities in performing daily living activities [2, 3]. Among the usable communication channels, hand gesture is very effective because its intuitiveness, and expressiveness to deliver information, even in noisy environments. As opposed to other cumbersome means of interaction, such as joysticks and sip-and-puff systems [4] which require users to physically manipulate controls or sensors, gesture based interfaces allow users to perform free hand and arm movements to control actuated devices using customized gestures. This feature is especially meaningful for individuals with high level spinal cord injuries who cannot perform hand and arm gestures dexterously. Gesture-based interaction is a promising alternative or complement to the existing control modalities.

In this paper, the problem of hands tracking and gesture recognition was addressed. Face and hand tracking was treated as an instance of the multi-object tracking (MOT) problem. The main focus in this paper is one hand tracking through interaction and occlusion. This

problem is challenging because hand gestures show variations among different users, while sharing a similar set of motion trajectories. Further, since each hand (left and right) have a similar appearance within the same individual, trackers can drift from one hand to the other when the hands are close together. In this paper, a combined approach is described which effectively addresses the challenging problem of tracking under self-occlusion.

The contribution of this paper is two-fold. First, it proposes an interaction model based on 3D particle filter tracking. Robust face and hand tracking performance was achieved by using 3D particle filters. The interaction model tackled the "false merge" and "false labeling" problems through hand interaction and occlusion. This model was divided into two parts, employing different features to solve these problems. Spatial and depth features were used to solve the "false merge" problem, while the motion feature was used to solve the "false labeling" problem. Second, the paper proposes a procedure to construct motion models and classify hand trajectories by utilizing the CONDENSATION algorithm. The combination of hand detection, tracking, and trajectory classification resulted in a real-time robust system for HRI in assistive technology.

The rest of the paper is organized as follows. Section 2 introduces the related work for the research in this paper. Section 3 provides the architecture of the proposed system with a brief introduction of each module. Section 4 discusses and describes in detail the proposed approaches for hand detection, tracking and gesture recognition. The comparative tests and results are presented in section 5. Section 6 gives the conclusions and discusses future work.

2 Related Work

Hand gesture recognition based interfaces typically encompass three modules: hands' segmentation, tracking, classification of the trajectories and hand configuration (the poses). A simple, yet common method for hand segmentation is to build color distribution models and back-project these models into unseen images. However, these methods may fail under non-fixed illumination and a cluttered background. Adding more modalities such as spatial and depth information may increase the reliability for hand segmentation under these constraints. This can be obtained by utilizing Time of Flight (TOF) cameras [5], stereo vision [6] or depth sensors, such as the Kinect® [7].

If the face and hands do not interact with each other, the tracking problem is a special case of the independent object tracking (IOT) problem which deals with non-rigid object tracking. If interaction or occlusion occurs, it becomes a special case of the MOT problem. Classical tracking approaches can be adopted to solve this problem if determining the shape of the hand is not required. For instance, particle filter [8] is a commonly used probabilistic based technique for object tracking. Perez et al. [9] enhanced object tracking under complex background by integrating an appearance model into the standard particle filter framework. Perez et al. also extended the particle filter framework to track multiple objects. Okuma et al. [10] incorporated a boost detector to extend the particle filter framework to deal with the MOT problem more effectively and enabled automatic initialization for potential multiple targets. One problem with these techniques is that they did not consider interaction among tracked objects as part of the particle filter framework. Tracking under interaction and occlusion conditions were studied in Kristan et al. [11] who integrated local motion information (calculated by optical flow) into a color-based particle filter framework. Kang et al. [12] tackled the ambiguity in the objects' location by registering video frames from multiple cameras. Khan et al. [13] tracked multiple interacting targets by analyzing their motion and adding a penalty function to the particle filter framework. Qu et al. [14] integrated a joint state space representation into a color-based particle filter and performed joint data association for multi-object tracking. A magnetic-inertia potential model was proposed to handle occluded tracking problems in a particle filtering framework [14]. Other tracking approaches that have been shown to be relatively successful for gesture tracking are CAMSHIFT [15] and CONDENSATION [16]. These techniques work for MOT problems under certain constraints, but may not always result in robust tracking for non-rigid objects in cluttered and unfixed environments with frequent occlusion.

Hidden Markov Models (HMM) are one of the most common approaches for gesture classification [17]. Common problems with HMM involve spotting the trajectories (gesture temporal segmentation) and selecting the right set of parameters for initialization of the filters. The CONDENSATION-based gesture trajectory recognition algorithm proposed by Black and Jepson [18] can be used to obtain more robust tracking and is less sensitive to parameter selection.

3 System Architecture

The architecture of the proposed system is illustrated in Fig. 1. The proposed hand gesture based recognition system consists of five modules: foreground segmentation, hand detection, tracking, gesture recognition, and an execution module. Each frame captured by the Kinect camera was passed as input to the system and then processed by each of the modules. Eight dynamic gestures were selected to constitute the gesture lexicon and then decoded as commands to control the robots. In addition, an execution module was set to control a TurtleBot™ mobile robot and a FANUC® robotic arm through the wireless network using TCP and Telnet Protocol respectively.

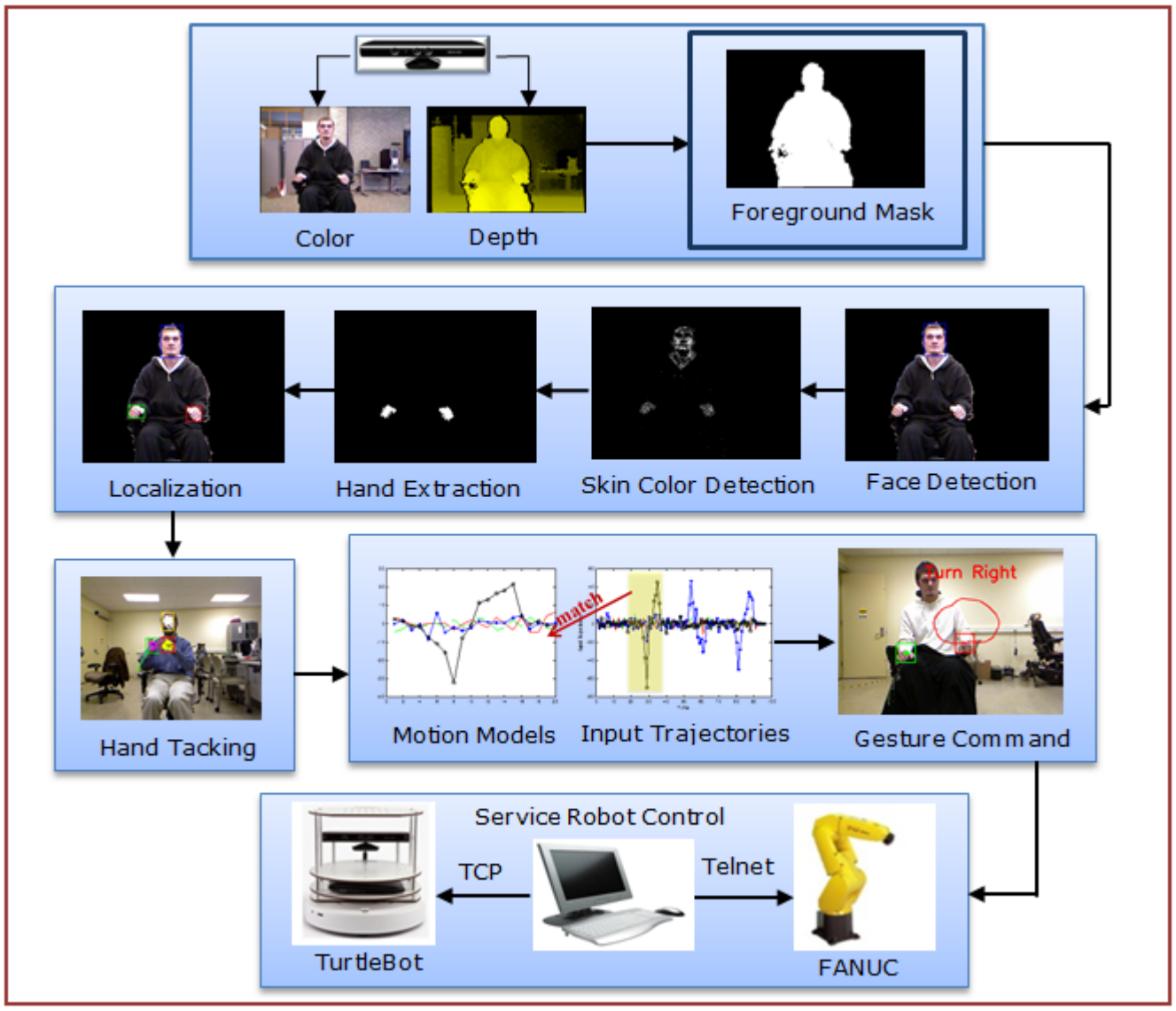


Fig. 1 System Overview

The modules mentioned above are briefly explained in the following subsections.

A. Foreground Segmentation Module

The goal of the foreground segmentation module is to reduce the search space of the hand detection algorithm. In this module, the background pixels were ruled out from the captured frames by using depth information and the whole human body was kept as the foreground.

B. Hand Detection Module

This module was used to provide an initialization region for the tracking algorithm. A skin and a non-skin color model were first created and then back-projected to the image in combination with morphological operations to obtain the face and hands blobs. The blobs of the hands were then extracted by excluding the face region with a face detector.

C. Hand Tracking Module

A 3D particle filter framework based on a color appearance model and depth information was used to track the face and hands through video sequences if no interaction or occlusion occurs. An interaction model based on spatial and motion information was incorporated into this particle filter framework to deal with occlusion. In addition, a neighborhood search algorithm was conducted to optimize the parameters of the interaction model.

D. Gesture Recognition Module

Hand tracking trajectories were segmented, compared with motion models and recognized by the CONDEN-

SATION method. These gestures were then mapped to a set of commands to control the robots.

E. Execution Module

The TurtleBot mobile robot was programmed using the Robotic Operation System (ROS), which contains a set of libraries developed by Willow Garage® for robotic control. The program is executed on the TurtleBot's workstation and waits for commands from the host computer sent through TCP Protocol. The FANUC robotic arm was programmed using KAREL, a scripted language used to control FANUC robots. The program is executed on the controller of the robotic arm through the Telnet Protocol.

4 Face and Hands Tracking

4.1 Foreground Segmentation

The goal of foreground segmentation in the proposed system was to reduce the complexity incurred in the hand detection phase. Initially, the connected components of the hands (the whole human body) were treated as the foreground. Two steps were employed to segment the human body from the cluttered and uncontrolled background (refer to Algorithm 4.1). The first step was to threshold the captured frame using depth information acquired from the Kinect camera. This involved keeping the pixels that were located within a certain range with respect to the camera, while discarding the rest. After applying connected components operations to those pixels, 'blobs' were obtained. The second step was to obtain a clean foreground by getting rid of small areas and keeping the largest blob as the foreground.

In the first step, the depth information was acquired by a Kinect sensor. For each pixel, the depth value was defined as D(i, j), where i and j denote the horizontal and vertical coordinates of the pixel. The distance of each pixel within an object from the depth sensor was mapped to intensity levels. Thus, the closer the object is to the sensor, the higher the intensity is (Fig. 2(a)). The image was thresholded by the depth value of each pixel. One adaptive threshold (T_{DH}) and an absolute threshold (T_{DL}) were set to rule out the pixels outside of this range. T_{DL} is set according to the technical specs provided for the Kinect sensor by Microsoft (usually it is set to 0.4 meter). T_{DH} is automatically set according to the depth value of the center region of the face (using face detector as in section 4.2). Only those pixels with a depth value between the two thresholds were kept in a binary mask image (Fig. 2(b)).

In the second step, the mask image was used to compute the area of the biggest region (blob), denoted as (A_{SH}). SH indicates the segmentation threshold for blob extraction. All the remaining blobs with a smaller area than A_{SH} were discarded (Fig. 2(c)). If the largest blob included an object that did not belong to the user's body, it would be discarded in a later stage because tracking is performed based on color, depth, spatial and motion information.

Algorithm 4.1 Foreground Segmentation Algorithm

INPUT: Low depth threshold T_{DL}; High depth threshold T_{DH}; Depth Image $D(i,j)$.

OUTPUT:

Pixel values of depth thresholding image $D_1(i,j)$;
Pixel value of hand mask image $D_2(i,j)$.

Begin

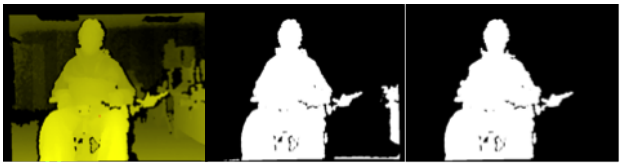
$$D_1(i,j) = \begin{cases} 1 & T_{DL} \le D(i,j) \le T_{DH} \\ 0 & otherwise \end{cases}$$

$//B_i$ is the ith blob in the mask image D_1

$A_{SH} = \max(\text{Area}(B_i))$

$$D_2(i,j) = \begin{cases} 1 & D(i,j) \in B_i \ \&\ \text{Area}(B_i) == A_{SH} \\ 0 & otherwise \end{cases}$$

End



(a) (b) (c)

Fig. 2 Foreground Segmentation. (a) Depth image; (b) Depth threshold mask; (c) Foreground segmentation mask.

4.2 Automatic Face and Hand Detection

The goal of this section was to detect the initial position of the face and hands. Skin and non-skin color histograms were constructed by using the Compaq database [19], which included more than 3000 images with skin color masks and more than 4000 non-skin color images. All the images were converted from RGB to the HSV color space. To obtain higher robustness for skin color detection, two 3D histograms were created−a skin and non-skin color histogram. The probability of a pixel being part of the hand was calculated as the ratio between the two histograms, the one belonging to the skin over the one belonging to the non-skin model (which was a proxy of the distinctiveness- the higher the ratio, the

more likely the two pixels belong to different color distributions). The mask image was obtained by applying the histogram ratio and back-projecting the probabilities of each pixel back into the image (Fig. 3(b)). To obtain the hand regions without the face, the region detected by a face detector [20] was removed from the target image. The remaining two largest blobs were considered as hand regions (Fig. 3(c), (d)). A geometrical moment was then used to obtain the centroids of the hands. Although the face and hand detection method mentioned above was effective, it was too slow for real-time processing. Therefore, this face and hands detection method was only used to provide automatic initialization for 3D particle filter tracking.

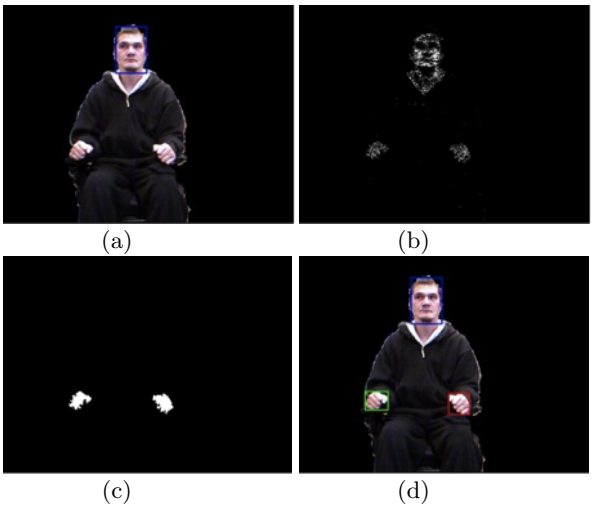


Fig. 3 Face and hand detection. (a) Face Detection; (b) Skin color detection; (c) Hand extraction; (d) Face and hand localization.

4.3 3D Particle Filter Tracking

A Sequential Importance Resampling (SIR) particle filter framework was used to track the face and hands through video sequences. See [9, 13, 21] for a detailed description of the SIR algorithm. Temporal filtering is given by equation (1):

$$p(X_t|Z_{1:t}) = k \cdot p(Z_t|X_t) \int p(X_t|X_{t-1}) p(X_{t-1}|Z_{1:t-1}) dx_{t-1} \tag{1}$$

where X_t is the state at time t, $Z_{1:t} = \{Z_1, \ldots, Z_t\}$ is all the observations from time 1 to t, $p(X_t|Z_{1:t})$ is the posterior distribution at time t, $p(Z_t|X_t)$ expresses the prior distribution (observation probability), $p(X_t|X_{t-1})$ is the transition probability that the system is at state X_t at time t given that the previous state is X_{t-1}, and k is a normalization factor. The posterior at time $t-1$ can be approximated by N weighted particles as $p(X_{t-1}|Z_{1:t-1}) \approx \{X^r_{t-1}, \omega^r_{t-1}\}^N_{r=1}$, where ω^r_{t-1} is the weight of the particle r at time $t-1$. Then for a given time t, N particles were propagated from the distribution with a transition model $p(X^r_t|X^r_{t-1})$ as $X_t \sim \sum_{r=1}^{N} \omega^r_{t-1} p(X^r_t|X^r_{t-1})$. Each sample was obtained by computing the product of its weight and its likelihood given that the observation $\omega^r_t \propto p(Z_t|X^r_t)$. This results in a weighted particle approximation as $p(X_t|Z_{1:t}) \approx \{X^r_t, \omega^r_t\}^N_{r=1}$. The tracker output can be approximated by the expectation $\widehat{X}_t \approx E[X_t|Z_{1:t}] = \sum_{r=1}^{N} \omega^r_t X^r_t$. Thus, equation (1) can be written as equation (2):

$$p(X_t|Z_{1:t}) \approx k \cdot p(Z_t|X_t) \sum_{r=1}^{N} \omega^r_t p(X^r_t|X^r_{t-1}) \tag{2}$$

The initial locations of the particles were selected in the face and hands centroids, each, which were calculated by using the method described in section 3.2.

The particle filtering process consists of three main stages: predicting, measuring and resampling. In the proposed system, for the prediction stage, a second order auto-regressive (AR) model [10, 21] was selected as the dynamic motion model as in equation (3):

$$X^r_t = A \cdot X^r_{t-1} + B \cdot (X^r_{t-1} - X^r_{t-2}) + \nu_t \tag{3}$$

Where A and B are set to 1 in this paper, ν_t is a Gaussian distribution with zero mean and variance matrix Σ, X^r_t is the state of the particle r at time t. Commonly, particle filters are applied to 2D tracking and only 2D locations and the scale are considered as the components for each state. The 2D particle filters demonstrate better performance in non-static and cluttered backgrounds than other tracking algorithms. However, 2D particle filters often fail, because the particles tend to reach a local minimum based on the information from the 2D images but not from the 3D real world coordinates. To tackle this problem, a 3D particle filter algorithm was employed. By integrating depth information into particle filters, it became much easier to distinguish two objects with similar color distribution. As a result, the state X^r_t in equation (3) can be written as $X^r_t = [x^r_t, y^r_t, z^r_t, s^r_t, x^r_{t-1}, y^r_{t-1}, z^r_{t-1}]$, where x^r_t, y^r_t, z^r_t are the 3D locations of particle r at time t and s_t is the scale of object at time t.

In the measuring stage of the particle filtering algorithm, the choice of the observation model plays a very important role in determining the weight of the particles. Color pre-processing can facilitate the extraction of the aforementioned features. Projection of a HSV skin color histogram is deemed as an efficient model for face and hands tracking. As explained earlier, in the initial phase, the face and hands were detected by the combination of depth-based threshold and image processing techniques.

The results were used to provide the initialization for the particle filters. Three reference HSV histogram models H_f^*, H_{h1}^* and H_{h2}^* were calculated for the face and hand regions, respectively, for tracking initialization. During the measuring phase, each particle, assigned in the predicting phase, was reweighted by the likelihood function based on the observations. For every hypothesized face or hand location (the region of a certain particle), the candidate histograms for face and hands of particle r were computed as H_f^r, H_{h1}^r and H_{h2}^r. A distance D is calculated from the Bhattacharyya similarity coefficient [9] as shown by equation (4).

$$D_i^r(H^*, H^r) = [1 - \Sigma\sqrt{H_i^* \cdot H_i^r}]^{1/2} \tag{4}$$

where $H_i^* = H_f^*, H_{h1}^*$ or H_{h2}^*. $H_i^r = H_f^r, H_{h1}^r$ or H_{h2}^r. The color-based observation likelihood function can be written as in equation (5):

$$p(Z_t|X_t) = k \cdot \exp(-\lambda_1 (D_i^r)^2) \tag{5}$$

where λ_1 is a measure of variance for the HSV histogram and k is the normalization factor to normalize the sum of all particles' weight to 1.

4.4 Hand Interaction and Occlusion

As mentioned in section 4.3, the color histogram based 3D particle filter framework was used for face and hands tracking. Three trackers were used to track the face and two hands separately. This approach was very effective for multiple independent objects tracking when the face and both hands did not interact or occlude each other. However, when the face and hand were in close proximity or the two hands interacted and occluded each other, the trackers may suffer from the "false merge" and "false labeling" problems [14]. The "false merge" problem means that the tracker loses track of the object being tracked and mistakenly focuses on a different object that has higher observation likelihood. Since the two hands are very similar in color and their shapes are changing while following certain trajectories, the "false merge" problem is very likely to happen during face and hands interaction. Conversely, the "false labeling" problem means the exchange of labels assigned to objects after interaction or occlusion occurs. This could happen for face and hands tracking when "false merge" does not occur and the tracker can be separated from each other after interaction.

When multiple objects interact with each other or occlusion occurred, the observations were not independent anymore. As a result, the posteriors of different objects were conditionally dependent. The "false merge" problem happens when the objects are conditionally dependent on each other. A first order Markov chain was used to analyze the dynamic interaction model between the face and both hands tracking. The correlation between the face and both hands when they interact with each other was represented as a Markov network (Fig. 4). If there was interaction between the face and both hands, the observation depended not only on its own state, but also on the observation of the interacting object (face or the other hand). For example, if the right hand is interacting with the left hand, the observation of the right hand would depend not only on its own state, but also on the left hand's observation. In this paper, an interaction model (CPMC) consisting of two sets of features were employed to solve "false merge" and "false labeling" problem. The first set of features including 3D spatial information was named "Competition Potential" (CP) and was used to keep the interacting occluded objects separated and to solve the "false merge" problem. The likelihood function for CP is defined as $\psi_1(X_{i,t}, X_{j,t})$.

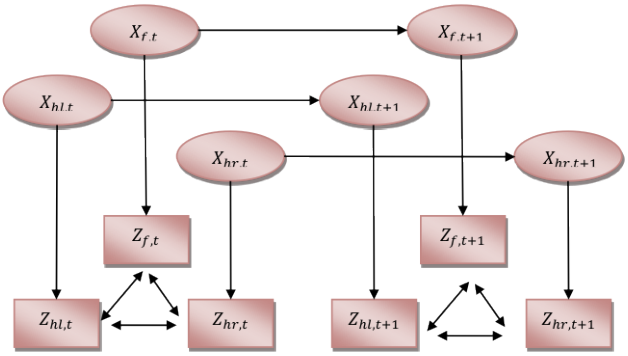


Fig. 4 Dynamic Markov network for face and both hand tracking. $x_{f,t}, x_{hr,t}, x_{hl,t}$, are the current hidden state for face, right and left hand. $x_{f,t+1}, x_{hr,t+1}, x_{hl,t+1}$, are the next hidden state for face, right and left hand. $z_{f,t}, z_{hr,t}, z_{hl,t}$, are the current observation of face, right and left hand. $z_{f,t+1}, z_{hr,t+1}, z_{hl,t+1}$ are the next frame's observation of face, right and left hand.

$$\begin{aligned}\psi_1(X_{i,t}, X_{j,t})^r = \beta_1 \cdot \exp(-\tfrac{\lambda_2}{d(X_{i,t}^r, X_{j,t})^2}) \\ \cdot \exp(-\lambda_3 d(X_{i,t}^r, X_{i,t-1})^2) \cdot \exp(-\tfrac{\lambda_4}{d_z(X_{i,t}^r, X_{j,t})^2})\end{aligned} \tag{6}$$

where i, j represents the interacted objects, $d(X_{i,t}^r, X_{j,t})$ is a 2D distance metric between two objects (e.g. Euclidian), $d(X_{i,t}^r, X_{i,t-1})$ is a distance metric between the current and previous position of object i, $d_z(X_{i,t}^r, X_{j,t})$ is the difference of depth value between two objects, β_1 is a normalization factor. These features were selected to solve the "false merge" problem for the following two reasons. Firstly, we want to incorporate the spatial and depth information into the likelihood function to weight each particle according to the competition potential between two objects. Here, the distance between the two objects in image plane and depth direction is a simple yet effective choice according to [13, 22]. Secondly, the object has potential to keep its inertial, so the distance

metric between the current and previous position was also concerned. In addition, since it is easier to specify the competition potential in the log domain, the exponential function was adopted.

The second set of features including 3D motion information was called "Motion Consistency" (MC). The likelihood function for this set of features is defined as $\psi_2(X_{i,t}, X_{j,t})$, which is based on the assumption that a particle region that has similar motion information to the previous state of that particle will have higher probability than a particle region that has distinct motion information. MC feature set was used to solve the "false labeling" problem. The equation similar as in [14] was employed to compute the likelihood function for the 3D motion features instead of using motion information from 2D image plane.

$$(\psi_2)^r_{i,t} = \beta_2 \cdot \exp(-\lambda_5(\theta^r_t)^2) \cdot \exp(-\lambda_6(A^r_t - A_{\text{ref},t})^2) \quad (7)$$

Where A^r_t is the norm of 3D motion vector of particle r at state t and $A_{\text{ref},t}$ is the reference motion vector, θ^r_t is the angle between the 3D motion vector of particle r and the reference vector. The 3D motion vector can be calculated by the difference of the current and previous 3D position vector. The reference motion vector can be calculated as the previous motion vector, β_2 is a normalization factor. These two features were selected because both angle and magnitude were required for motion to weight each particle and the 3D motion vector were used because they represented the 3D real movement of the face and hands. The exponential function was adopted to specify the likelihood function in the log domain.

If interaction happens, the observation likelihood function for particle r can be calculated as equation (8), which is a combination of equation (5), (6) and (7):

$$p(Z_t|X_t) = \beta \cdot \exp(-\lambda_1(D^r_i)^2) \cdot \psi_1(X_{i,t}, X_{j,t})^r \cdot (\psi_2)^r_{i,t} \quad (8)$$

where β is a normalization factor. By using the characteristic of the exponential function, equation (9) is obtained:

$$\begin{aligned} p(Z_t|X_t) = \beta \cdot \exp\{-[&\lambda_1(D^r_i)^2 + \tfrac{\lambda_2}{d(X^r_{i,t}, X_{j,t})^2} \\ &+\lambda_3 d(X^r_{i,t}, X_{i,t-1})^2 + \tfrac{\lambda_4}{d_z(X^r_{i,t}, X_{j,t})^2} \\ &+\lambda_5(\theta^r_t)^2 + \lambda_6(A^r_t - A_{\text{ref},t})^2]\} \end{aligned} \quad (9)$$

When the objects did not interact with each other, the approach performed as multiple independent trackers by using a color based 3D particle filter framework. Equation (5) was used to compute the observation probability of particle r. However, when the objects interacted (e.g. partial or complete occlusion occurred), spatial, depth and motion information were taken into account together to solve the "false merge" and "false labeling" problems. Equation (9) was used to compute the observation probability of particle r at this condition. Face and hand tracking results with interaction were shown in Fig. 5. The algorithm for hand tracking during interaction and occlusion was shown in Algorithm 4.2.

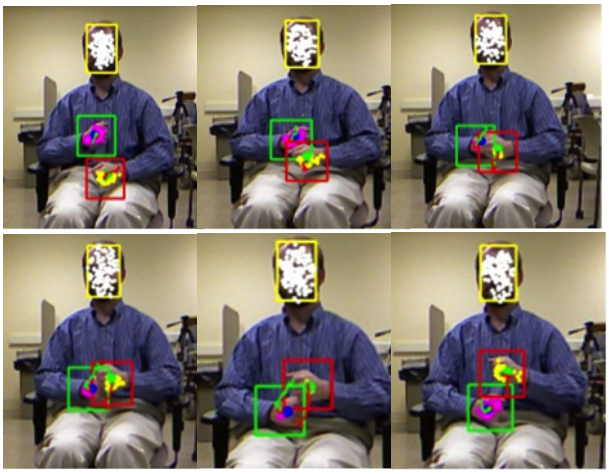


Fig. 5 Hand tracking through interaction and occlusion.

Algorithm 4.2 3D particle filter tracking with interaction model

INPUT: Reference HSV histogram models H^*_f, H^*_{h1} and H^*_{h2};

OUTPUT: Centroids of the face and both hands and the associated bounding box.

Begin

1: **Initialize:**

//Initialize particle states and weight for face and both hands as:

$x^i_0 = x^*_0,$

$\omega^i_0 = \frac{1}{n}$ for $i = 1, \ldots, n;$

2: **Predict, Measure and Resample:**

for $i = 1, 2, 3$ *//(1-face, 2-right hand, 3-left hand)*

for $r = 1$ to N

//N is the number of particles

//compute transition model for 3D particle filters

$X^r_t = X^r_{t-1} + (X^r_{t-1} - X^r_{t-2}) + \nu_t$

//Compute candidate histograms H^r

$D_i(H^*, H^r) = [1 - \Sigma\sqrt{H^* \cdot H^r}]^{\frac{1}{2}}$

//Calculate the weight:

$\omega^r_{i,t} = k \cdot \exp(-\lambda D^2_{i,t})$

//Check interaction

if interaction happens for object i and j

for $q = 1$ to N

//computer interaction likelihood ψ_1 *and* ψ_2*:*

use equation (6) to compute $(\psi_1)^q_{i,t}(X_{i,t}, X_{j,t})$

use equation (7) to compute $(\psi_2)^q_{i,t}$

//Calculate the weight:

$\omega^q_{i,t} = \omega^q_{i,t} \cdot (\psi_1)^q_{i,t} \cdot (\psi_2)^q_{i,t}$

end for

end if

end for

Normalize the weights of particles for each object.

Reset distribution of particles according to their weight.

Estimate $\widehat{X}_{i,t} = \sum_{r=1}^{N} \omega^r_{i,t} X^r_{i,t}$

end for

End

4.5 Parameters Search for Interaction Model

As mentioned in section 4.4, if interaction occurs between two objects, a interaction model was applied to the color based 3D particle filter framework to solve "false merge" and "false labeling" problem for face and hands. However, from equation (9), we can see a set of parameters need to be established to calculate the weight for each particle. Manually setting is a tedious ad-hoc method which does not guarantee optimality. To achieve better tracking performance, a neighborhood search method was employed to find the optimal parameters. Let p be the parameter vector: $p = [\lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5, \lambda_6]$. During the parameter search process, each parameter was incremented by a positive and negative margin. This method was initialized with a random solution. The parameters search process is described as in Fig. 6. An iteration in the neighborhood search algorithm is defined as a full round of increment and decrement of the parameter vector, which each is applied to tracking the hands in all the frames in the training video sequences and obtaining for each a corresponding accuracy. Then, the parameter vector which had assigned the highest accuracy value is selected as the optimal parameter for that round (iteration). The parameter search process ends when the maximum iteration number was reached or the parameters stoped to change after two continuous iterations. An example of the parameter search processes is shown in Fig. 7. Five seeds were randomly selected to initialize each searching process. An optimal set of parameter was then obtained for each initial seed by the neighborhood search algorithm.

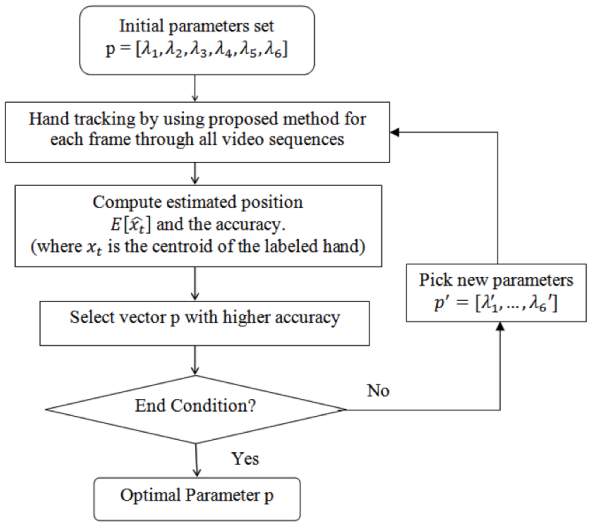


Fig. 6 Optimal parameter search for interaction model

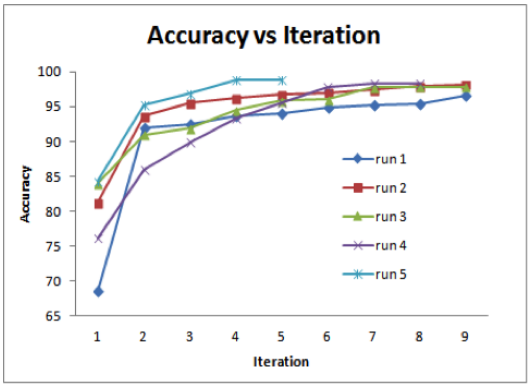


Fig. 7 Accuracy vs. Iteration

5 Gesture Recognition

The positions of the hands in each frame of the video sequences were acquired in the tracking stage. The trajectories constituted by these acquired discrete locations were then compared with the motion model of each gesture in the lexicon. These motion models were constructed by using the training data collected from eight able-bodied subjects. Although in different instances, the trajectories for each gesture collected from different subjects or even different instances of the same subject may look similar, the precise duration of each sub-trajectory within the trajectory were different. Dynamic time warping (DTW) algorithm [23] was employed to align all the trajectories for each motion model of the gesture in the lexicon. The velocity components in the horizontal and vertical directions for both hands were adopted to compose the feature elements for each motion model [21]. The following procedure was adopted to obtain the motion models (Algorithm 5.1).

The CONDENSATION algorithm [18] was used to recognize hand gestures in the lexicon. The CONDENSATION algorithm applies the random sampling to model probability density functions. Instead of fitting the observed data to a specific equation, the CONDENSATION algorithm uses a set of weighted samples to approximate the observed data. There are four modules in the CONDENSATION algorithm to recognize dynamic hand gestures: initialization, prediction, updating and classification. The original algorithm was extended to work for two hands. For the original equation in [18], the state at time t can be described as $S_t = (\mu, \phi, \alpha, \rho)$. For this paper, this expression was extended to:

$$\begin{aligned} S_t &= (\mu, \phi^i, \alpha^i, \rho^i) \\ &= (\mu, \phi^{\text{right}}, \phi^{\text{left}}, \alpha^{\text{right}}, \alpha^{\text{left}}, \rho^{\text{right}}, \rho^{\text{left}}) \end{aligned} \tag{10}$$

where, μ is the index of the motion models, ϕ is the current phase in the model, α is an amplitude scaling factor, $i \in \{\text{right hand, left hand}\}$, ρ is a time dimension scaling factor.

Algorithm 5.1 Procedures to Construct Motion Models

INPUT: Number of gestures G; number of subjects S; number of sampling trajectories from each subject T; horizontal and vertical velocity for left and right hand $V_p^m, m = 1, \ldots, S \times T$.

OUTPUT: Motion models for each gesture.

```
Begin
for k = 1 : G
    for j = 1 : S
        for i = 1 : T − 1
            Align V_p^i with V_p^{i+1} to obtain V_ap^i
        end for
        Align V_p^T with V_p^1 to obtain V_ap^T
        V_p^j = Σ_{i=1}^{T} V_ap^i / T
    end for
    for j = 1 : S − 1
        Align V_p^j with V_p^{j+1} to obtain V_ap^j
    end for
    Align V_p^S with V_p^1 to obtain V_ap^S
    V_p^k = Σ_{i=1}^{S} V_ap^i / S
end for
End
```

6 Experiments and Results

6.1 Hand Tracking Performance Evaluation

A dataset consisting of 16 videos (4 subject x 4 activities) was used to test the new tracking algorithm. The total number of frames for these videos was 6160, along with hand labeled ground truth data for both the right and left hand. The activities performed by the subjects were: (a) holding a cup with two hands (1010 frames), (b) clapping hands (2230 frames), (c) moving hands up and down (one hand occludes the other) (1320 frames), (d) rotating hands forward and backward (two hands occlude each other) (1600 frames). The videos were captured using a Kinect camera at 30Hz with an image size of 640 x 480. The tracking performance for the proposed method (3DCPMC) was compared to Kinect® OpenNI SDK Skeleton tracking algorithm and three other existing state of the art tracking algorithms: (i) mean shift (MS), (ii) standard particle filter (SPF) tracking, and (iii) Markov Chain Monte Carlo (MCMC) based particle filter tracking. All the algorithms used to compare our approach were not optimized to deal with interaction and occlusion.

Although both face and two hands were tracked in the proposed system, the focus was on hands tracking through interaction and occlusion. Thus, only the performance of both hands tracking was evaluated for all the algorithms. This evaluation was performed between the ground truth (GT) and system tracker (ST). To obtain better evaluation, a set of metrics presented in [25, 26, 27] was employed.

A. Object Tracking Error (OTE)

The object tracking error is defined as the average discrepancy between the GT bounding box centroid and the centroid of ST:

$$OTE = \frac{1}{M}\sum_{i=1}^{M}\sqrt{(x_i^g - x_i^r)^2 + (y_i^g - y_i^r)^2} \tag{11}$$

where M represents the total number of tracked frames, (x_i^g, y_i^g) represents the ground truth coordinate of the object's centroid in ith frame, and (x_i^r, y_i^r) represents the tracked centroid of the object in ith frame.

B. Mean spatial overlap ratio (MSOR):

$$MSOR(GT_i, ST_i) = \frac{1}{M}\sum_{i=1}^{M}\frac{\text{Area}(GT_i \cap ST_i)}{\text{Area}(GT_i \cup ST_i)} \tag{12}$$

where GT_i is the Ground truth bounding box in ith frame and ST_i.

C. Accuracy:

$$\text{Accuracy} = \frac{\text{total number of true positives and true negative}}{\text{total number of ground truth points}} \tag{13}$$

where the true positive is defined as the number of frames that the objects exists and the tracker can find the object correctly. While the true negative means the number of frames that the object does not exist and tracker also agrees that the object is absent [27].

D. False Merge Ratio (FMR):

The "false merge" was defined when the tracker of one hand occupies 80% of the space corresponding to the other hand. FMR is defined as the ratio between the number of "false merge" frames and the total number of frames.

E. False Labeling (FL):

The "false labeling" was defined when the trackers of both hands were labeled incorrectly during/after interaction or occlusion.

The optimal parameter set resulting from neighborhood search algorithm was applied to the interaction model. The metrics mentioned above were computed by using the tracking results (centroid and bounding box). Table 1 shows the value of each metrics under different algorithms. From the table, the proposed algorithm (3DCPMC) exhibits the best performance for interaction and occlusion conditions through all the frames in the testing videos.

Our dataset included four activities where hands interaction and occlusion happened quite frequently. The tracking sequences of "clapping hands", "holding a cup" , and "rotating hands forward and backward" activities are shown in Fig. 8, 9, and 10, respectively. An example of "moving hands up and down" activity can be seen in Fig. 5. These video sequences demonstrated how the tracker worked under hand interaction and occlusion conditions.

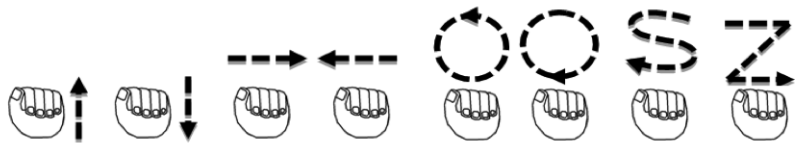


Fig. 11 Gesture lexicon, left to right: upward, downward, rightward, leftward, clockwise circle, anti-clock circle, S, Z.

Fig. 12 Dynamic motion models constructed by using DTW.

Table 1 Tracking performance and computational cost

Metric Method	Kinect®	MS	SPF	MCMC	3DCPMC (proposed)
Accuracy (%)	32.60	20.46	67.08	72.34	98.81
Mean Spatial Overlap Ratio	0.2406	0.0849	0.7087	0.5245	0.7670
Object Tracking Error (pixel)	24.3723	19.9022	7.8077	7.3930	5.7711
False Merge Ratio	0	0.1008	0.6714	0.3887	0
False Labeling (110 interactions)	3	13	15	6	2
Particle Number	–	–	100	100	100
Computational Cost (s/frame)	0.033	0.036	0.039	0.039	0.041

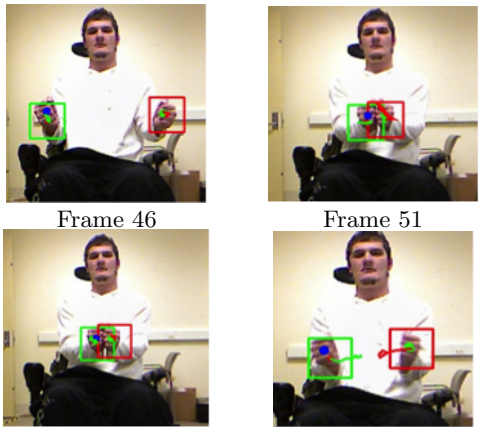


Fig. 8 "Clapping Hands" Activity

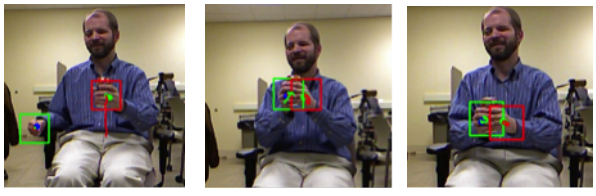


Fig. 9 "Holding a Cup" Activity

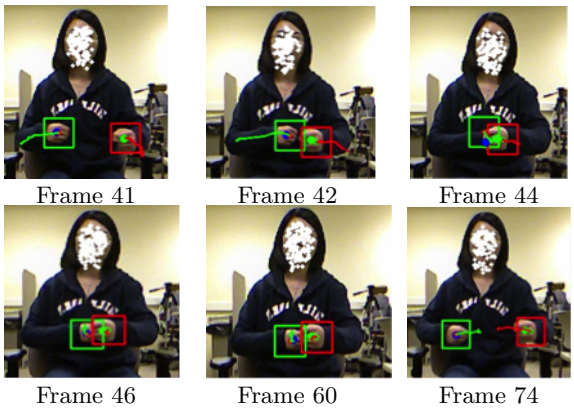


Fig. 10 "Rotating hands forward and backward" Activity

6.2 Recognition Accuracy

For this proposed system (as in Fig.11). an eight-gesture lexicon was determined through interviews with individuals with upper extremity mobility impairments and utilizing prior data analysis results [24]. The size of the lexicon was determined by minimizing the memory load for the user and at the same time satisfying the minimum command requirements for robotic control. The spatio-temporal trajectories of those gestures were recognized by the proposed system and, in turn, sent as commands to control the robots. The motion model for each gesture trajectory was constructed by using the DTW algorithm. The horizontal and vertical velocities of right and left hands were used as the main feature components. The motion models constructed for this eight-gesture lexicon are shown in Fig. 12 (the red and cyan lines showed the horizontal and vertical velocities of right hand, while the blue and dark lines showed horizontal and vertical velocity of the left hand). These plots were the templates showed the movement of each hand for each gesture.

The recognition accuracy of the proposed system was validated by eight able-bodied subjects. Ten sessions were used for cross validation for each gesture (k-fold with k=10). In each session 72 observations (8 gestures x 9 repetitions) were used for training and 8 observations were used for testing. This cross validation resulted in an average accuracy of 97.5%. A confusion matrix was computed and is shown in Fig. 13 (with a temporal window size of w = 20). The receiver operating characteristic (ROC) curve displayed the system performance when changing the size of the window from 10 to 24 to different values (as shown by Fig. 14). All the parameters for gesture recognition were tuned under controlled laboratory conditions. Readjusting parameters for non-laboratory

environment is challenging and will be the focus of our future work.

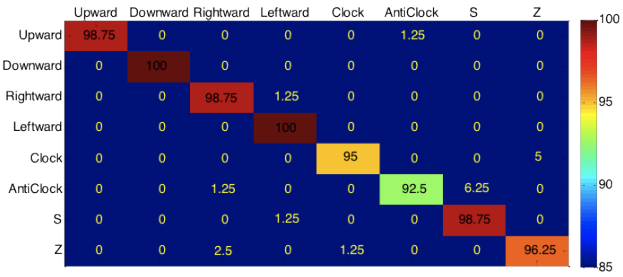


Fig. 13 Confusion matrix with window size w =20.

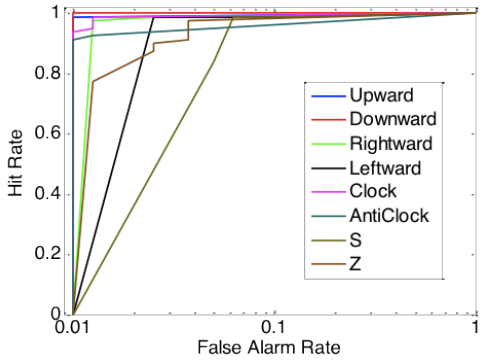
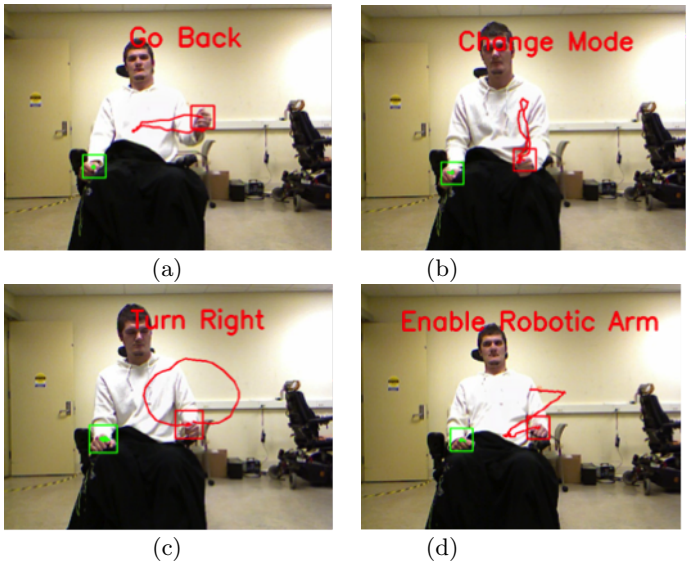


Fig. 14 ROC Curve showing recognition performance for each gesture.

6.3 Laboratory Experiment

A simulated laboratory task was performed by five subjects including three able-bodied individuals and two individuals with quadriplegia due to a cervical spinal cord injury. Each of the subjects completed the task five times by controlling two robots working cooperatively. Both robots were controlled using hand gestures and the recognition system issued the commands to control the robots. While the mobile robot cannot present any harm to the user, the robotic arm is of industrial type, and therefore a minimum distance must be respected to assure proper safety to the user. While the task completion time may differ according to the skill level of the subjects performing the experiment, we believe they reflect general trends in completion times of a large pool of users with spinal cord injuries. In this experiment, the subject navigated a mobile robot to a position near a robotic arm and activated the robotic arm to add a reagent to a beaker. The gestures from the lexicon (as in Fig. 11) were employed to control the mobile robot and the robotic arm and then mapped to the commands: 'change mode', 'robotic arm action', 'go forward', 'go back', 'turn left', 'turn right', 'stop' and 'enable robotic arm'. Fig. 15 shows four examples of gestures mapping when the system was used by an subject with upper-level spinal cord injury to control the mobile robot and the robotic arm. Three modes were used to control the mobile robot: discrete, continuous and hybrid mode (discrete plus continuous mode). In discrete mode, the robot moved a fixed increment of distance, every time that a command was issued. While in continuous mode, the robot responded to a given command, until the 'stop' or another command was issued. One distinctive gesture command ('upward') was utilized to switch between the discrete and continuous modes. In the experiment, the three control modes were used by each subject to control the mobile robot and to performed the task five times. The resulting average task completion time including performance of able-bodied subjects and subjects with quadriplegia were 241.8, 134.7, and 169.6 seconds, for the discrete, continuous and hybrid, respectively.



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

Fig. 15 Gesture mapping for robotic control

6.4 Real-Time Performance

The proposed system demostrated real-time performance during hand detection, tracking, recognition and robotic control. This system was implemented by using Microsoft Visual Studio C++ 2010 and OpenCV Library 2.1. The whole program was run on a 3rd Gen Intel CoreTM i3-3220 PC (Dual Core, 3.30GHz 3MB, w/HD2500 Graphics). The computational cost for detection, tracking, recognition, and robotic control were: 0.052s, 0.041s, 0.001s,

and 0.003s respectively. Among all the modules, the hand detection module was only performed when a user started to use the system. Thus, the total processing time for a frame after initialization was 0.045s.

7 Conclusions and Future Work

A real time hand gesture recognition based interface was developed for individuals with spinal cord injuries. A 3D particle filter was used to track both hands through occlusion. An accuracy of 98.81% was obtained by using the optimal parameters for tracking. A comparison between our proposed approach (3DCPMC) and Kinect skeleton tracking with the other three state of the art algorithms demonstrated that our approach can achieve robust performance for hand tracking through interaction and occlusion conditions. Both "false merge" and "false labeling" problems were tackled by the proposed method. Computation cost table demonstrates the real time performance of the proposed tracking algorithm. A training procedure was proposed to obtain motion models for each gesture in the lexicon, and the CONDENSATION algorithm was used to classify bimanual gesture with a recognition accuracy of 97.5%.

A simulated laboratory task test was conducted using a mobile robot in concert with a robotic arm through the gesture recognition based interface proposed in this paper. It was found that the gesture recognition algorithm was robust enough to support the completion of this task. Three control modes (discrete, continuous, and hybrid) were compared. Results showed that the continuous mode required the least average task completion time, while the discrete control mode required the longest. Because the discrete mode was more labor-intensive, fatigue may have been a factor for the quadriplegic subjects. Therefore, the authors recommend using continuous control mode in general, and using discrete mode only when the robot is near the target for fine adjustment and manipulation. Real-time performance was required to ensure a responsive user system. Future work includes: (1) develop more effective and robust algorithms to obtain higher tracking accuracy; and (2) extend the system to be used in more laboratory tasks.

References

[1] Jacko, J.A. (2011) Human-Computer Interaction Design and Development Approaches. In: *14th HCI International Conference*, pp. 169-180.

[2] Moon, I., Lee, M., Ryu, J., and Mun, M. (2003) Intelligent Robotic Wheelchair with EMG-, Gesture-, and Voice-based Interfaces. In: *International Conference on Intelligent Robots and Systems*, IEEE Press, pp. 3453-3458.

[3] Reale, M., Liu, P. and Yin. L.J. (2011) Using eye gaze, head pose and facial expression for personalized non-player character interaction. In: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, IEEE Press, pp. 13-18.

[4] Huo, X., and Ghovanloo, M. (2009) Using Unconstrained Tongue Motion as an Alternative Control Mechanism for Wheeled Mobility, *IEEE Transaction on Biomedical Engineering*, pp. 1719-1726.

[5] Goektuerk, B. S. and Tomasi, C. (2004) 3d head tracking based on recognition and interpolation using a time-of flight depth sensor. *IEEE conference on computer vision and pattern recognition*, pp. 211-217.

[6] Li, Z., Jarvis, R. (2009) A multi-modal gesture recognition system in a Human-robot interaction scenario. In: *Proceedings of the IEEE International workshop on robotic and sensors environments*, pp. 41-46.

[7] Suma, EA., Lange, B., Rizzo, A., Krum, DM., and Bolas, M. (2011) FAAST: The Flexible Action and Articulated Skeleton Toolkit. *IEEE Virtual Reality*, pp. 247-248.

[8] Maskell, S., Gordon, N., and Clapp, T. (2002) A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *IEEE transaction on signal processing*, pp. 174-188.

[9] Perez, P., Hue, C., Vermaak, J., and Gangnet, M. (2002) Color-based probabilistic tracking, *LNCS*, Springer, Heidelberg, Vol: 2350, pp. 661-675.

[10] Okuma, K., Taleghani, A., Freitas, N., Little, J.J. and Lowe, D.G. (2004) A boosted particle filter: multitarget detection and tracking. *ECCV*, pp. 28-39.

[11] Kristan, M., Pers, J., Kovacic, S., and Leonardis, A. (2009) A local-motion-based probabilistic model for visual tracking. *Pattern Recognition*, pp. 5-28.

[12] Kang, J. (2003) Continuous tracking within and across camera streams. *Computer vision and pattern recognition*, Vol: 1, pp:267-272.

[13] Khan, Z., Balch, T., and Dellaert, F. (2004) An MCMC-based particle filter for tracking multiple interacting targets. *ECCV*, pp. 279-290.

[14] Qu, W., Schonfeld, D., and Mohamed, M. (2007) Real-time distributed multi-object tracking using multiple interactive trackers and a magnetic-inertia potential model. *IEEE transactions on Multimedia*, Vol: 9(3), pp. 511-519.

[15] Bradski, G.R. (1998) Computer vision face tracking as a component of a perceptual user interface. In: *Workshop on applications of computer vision*, pp. 214-219.

[16] Isard, M., Black, A. (1998) CONDENSATION: Conditional density propagation for visual tracking. *International Journal of Computer Vision*, Vol. 29, pp. 5-28.

[17] Bilal, S., Akmeliawati, R., Shafie, A.A., and Salami, M.J.E. (2011) Hidden Markov Model for human to computer interaction: a study on human hand gesture recognition, *Artificial Intelligence*, pp. 1-22.

[18] Black, M.J., and Jepson, A.D. (1998) A probabilistic framework for matching temporal trajectories: CONDENSATION-based recognition of gesture and expressions. *ECCV*, pp. 909-924.

[19] M. J. Jones and J. M. Rehg, Statistical color models with application to skin detection, *IEEE Computer Society Conference on computer vision and pattern recognition*, vol.1, pp. 81-96.

[20] Viola, P., and Jones, M. (2001) Rapid object detection using a boosted cascade of simple features. In: *International Conference on Computer Vision and Pattern Recognition*, pp. 511-518.

[21] Hess, R., Fern, A. (2009) Discriminatively Trained Particle Filters for Complex Multi-Object Tracking. In: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 240-247.

[22] Yu, T. and Wu, Y. (2004) Collaborative tracking of multiple targets. *Computer society conference on computer vision and pattern recognition (CVPR)*, pp. 834-841.

[23] Aach J. and Church G.M. (2001) Alignment gene expression time series with time warping algorithms, *Bioinformatics*, Oxford University Press, Vol: 17(6), pp. 495-508.

[24] Jiang, H., Wachs, J.P., and Duerstock, B.S. (2012) Facilitated Gesture Recognition Based Interfaces for People with Upper Extremity Physical Impairments. *Lecture Notes in Computer Science*, 2012, Volume 7441, Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications, pp. 228-235.

[25] Black, J., Ellis, T., and Rosin, P. (2005) A Noval Method for Video Tracking Performance Evaluation. *The joint IEEE International Workshop on VS-PETS*, Beijing, October, pp. 15-16.

[26] Ellis, T. (2002) Performance Metrics and Methods for Tracking in Surveillance, *Third IEEE International Workshop on Performance Evaluation of Tracking and Surveillance*, June, Copenhagen, Denmark, pp. 26-31

[27] Pingali, G., and Segen, J., Performance evaluation of people tracking systems, In: *Proc. IEEE Workshop on Application of Computer Vision*, pp. 33-38.

[28] Pan, P., Porikli, F., and Schonfeld, D. (2009) A new method for trackning performance evaluation based on a reflective model and perturbation analysis. *IEEE International Conference on Acoustics, speech and signal processing*, pp. 3529-3532.