Tracking Multiple Objects and Action Likelihoods

Chi-Min Oh and Chil-Woo Lee

Abstract. In this paper we propose a method which can improve MRF-Particle filters used to solve the hijacking problem; independent particle filters for tracking each object can be kidnapped by a neighboring target which has higher likelihood than that of real target. In the method the motion model built by Markov random field (MRF) has been usefully applied for avoiding hijacking by lowering the weight of particles which are very close to any neighboring target. The MRF unary and pairwise potential functions of neighboring targets are defined as the penalty function to lower the particle's weights. And potential function can be reused for defining action likelihood which can measure the motion of object group.

Keywords: Multiple Object Tracking, Action Likelihood, Gesture Recognition.

1 Introduction

Data association for tracking multiple objects has been an important issue because the maintenance of the list of the detected objects is needed in the research fields of image and signal processing, computer vision, stochastic process and pattern recognition. The representative applications of tracking multiple objects is such as visual player tracking in soccer broadcast system, visual surveillance in building or with mobile robots, automobile's collision avoidance against pedestrians and bicycles, multiple touch tracking for mobile devices, activity analysis of insects or people, video compression, and human-computer interaction.

The process of data association is like the process of multiple object tracking that assigns proper identities to the detected objects in the consecutive video

Chi-Min Oh · Chil-Woo Lee
School of Electronics and Computer Engineering,
Chonnam National University, Gwangju, Korea
e-mail: sapeyes@image.chonnam.ac.kr, leecw@chonnam.ac.kr

images. By only using object detectors such as a labeling method on object segmentation image not considering sequential consistency of object identities, the list of detected objects changes at every frame. There must be frequently and repeatedly appearing objects and it is required to assign same identities to these sequentially existing objects. Multiple hypothesis tracking [1] and the joint probabilistic data association filter (JSDAF) [2] are the basic frameworks in this data association area. The multiple hypothesis tracker and JPDAF with a particle filter [3] show some potentialities for nonlinear systems. However, the hijacking problem has not been explicitly resolved. Some related works [1,3,6,7] focused on other aspects, such as the observation model and feature descriptors.

Khan et al. [4] first have attempted to explicitly represent the hijacking problem. Khan uses a graph representation of neighboring objects since hijacking happened when the locations of the objects were very close. Considering the distance between objects, neighboring objects are connected as graph edges. This graph can be assumed as Markov random field (MRF) where the properties of neighboring objects can be defined as the potential functions and exploited for object tracking problem. Using MRF, a description of neighboring objects, Khan defined MRF motion model in Markov chain Monte Carlo-based particle filter and solved hijacking problem. Based on MRF motion model, to solve the hijacking problem, Khan used a penalty function from MRF pairwise potential functions to lower the particle weight nearly approached to neighbors than object itself.

This paper basically adapts Khan's MRF motion model in our multiple particle filters which we call here as MRF-Particle filters. The pairwise term of MRF-Particle filter for the penalty function is similar to Khan's method but we try to additionally utilize the unary term because the graphs of neighboring objects are built at every end of frame for penalty function in next frame that means the actual penalty function is not made at current time. Therefore unary term could compensate the weakness of pairwise term which is based on the outdated graph by one time step. Unary term can be used in penalty function with current observation image; we can add current information to one-frame outdated graph for hijacking problem.

MRF graphs represent the details of movements of all objects. Based on the potential functions of MRF graphs, it is possible to know the motion patterns of multiple objects. We present how to define the action likelihoods of neighboring multiple targets. The action likelihoods represent the motion and chords of touched fingertips based on MRF potential functions. The motion and chord likelihoods are such as clues for translation, rotation, scaling motions and structure of multiple touch points. These likelihoods can be estimated concurrently as a joint likelihood to define more complex gesture recognition. In applications, touch command can benefit from this definition strategy. By multiplying any action likelihood as a joint likelihood [9] it is possible to generate the gesture recognition command.

The rest of this paper is organized as follows. Section 2 outlines from the object detection to data association of multiple particle filters. Section 3 describes how to build MRF graphs of multiple objects. Based on MRF, the penalty function with pairwise and unary terms can help to avoid the hijacking problem. Section 4 explains the action likelihoods of the motion and structure of grouped object neighbors. Section 5 concludes the paper and discusses about future research directions.

2 Particle Filters for Tracking Multiple Targets

For the simulation of tracking multiple objects, we detect dynamically moving multiple touch points in our tabletop display system as shown in Fig. 1. The touched fingertips are obtained through an infrared camera capturing the scattered infrared rays from the acrylic display [8]. Using the segmented binary image of fingertips we detect the touch points using 8-neighbor labeling algorithm. The result of detection is the list of ID, location and size of each touch point but the ID of each touch point is not guaranteed to be same in upcoming image frames [5].

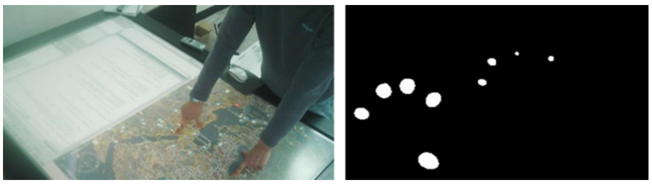


Fig. 1 Simulation system of object detection of touch points based on our tabletop display

Fig. 2 shows that the identities of detected objects change within two consecutive frames. Using a labeling method, it is not possible to assign proper identities to objects consistently in consecutive image frames. The data association of detected objects maintains the identities of multiple touch points throughout a video sequence. The easiest way of data association simply attaches a tracker to each target. In this paper, we prefer to use independent particle filters for tracking each object since particle filter is very robust in nonlinear and dynamic environment. For tracking each object identity, we follow the process of particle filtering.

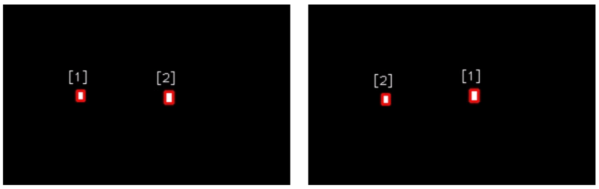


Fig. 2 The identities of detected objects using 8-neighbor labeling method can change

Particle filter is a implementation version of Bayesian filter [3], which estimates the posterior distribution by updating previous posterior distribution with discrete samples (particles). The benefit of Bayesian filter is the recyclability of previous result by adding current likelihood and prior into the previous posterior as like this

$$p(x_t|Y_t) = \alpha p(y_t|x_t) \int p(x_t|x_{t-1}) p(x_{t-1}|Y_{t-1})$$

where, x_t is the object state (location) and $Y_t = \{Y_1, \dots, Y_t\}$ is the vector of observations which are the consecutive segmented binary images. The current posterior with all observation Y_t is updated from previous posterior $\mathrm{p}(x_{t-1}|Y_{t-1})$ using prior $\mathrm{p}(x_t|x_{t-1})$ and likelihood $\mathrm{p}(y_t|x_t)$.

Particle filter has the prediction, evaluation and resampling steps. In prediction step every particle is predicted from previous particle set as like

$$p(x_{i,t}|Y_{i,t}) \approx \left[x_{i,t}^{(s)}, w_{i,t}^{(s)}\right]_{s=1}^{N}$$

where, $x_{i,t}^{(s)}$ is a particle, $w_{i,t}^{(s)}$ is the particle's weight and *N* is the number of particles.

The particles are proposed by below proposal distribution (transition model)

$$x_{i,t}^{(s)} \sim p(x_{i,t}|x_{i,t-1})$$

In evaluation step, every particle's weight is determined by its likelihood to update the predicted posterior to current time:

$$w_{i,t}^{(s)} = p(y_t|x_{i,t}^{(s)})$$

After updating the posterior from previous one, each target's position can be estimated by particle set of *i*th particle filter. The expected value $\overline{x_{\iota,t}}$ of the particle set in *i*th particle filter statistically gives a reliable and smooth position of *i*th object.

$$\overline{x_{\iota,t}} = \sum_{s=1}^{N} w_{i,t}^{(s)} x_{it}^{(s)}$$

For estimation of likelihood, we use the integral image of segmentation image which is shown in Fig. 3. The likelihood is related to the number of white pixels covered by the particle window. Therefore the number of counting those pixels in integral image reduces to 4 times. Then the likelihood using integral image $\grave{y}_t$ is

$$w_{i,t}^{(s)} = \beta\left(\grave{y}_t(x_{i,t}^{(s)}:1) - \grave{y}_t(x_{i,t}^{(s)}:2) - \grave{y}_t(x_{i,t}^{(s)}:3) + \grave{y}_t(x_{i,t}^{(s)}:4)\right)$$

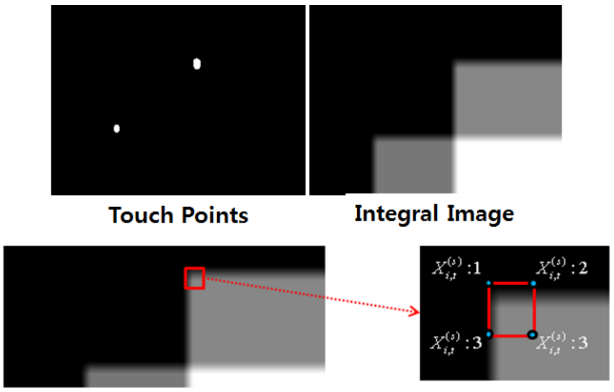


Fig. 3 The observation image (binary) is transformed to the integral image to calculate particle weights fast. The particle weight is the likelihood which is relative to the amount of white pixels within the location of particle which appearance can be seen here as a rectangle.

where, $\grave{y}_t(x_{i,t}^{(s)}:1)$, $\grave{y}_t(x_{i,t}^{(s)}:2)$, $\grave{y}_t(x_{i,t}^{(s)}:3)$, and $\grave{y}_t(x_{i,t}^{(s)}:4)$ are pixel values at the apexes of the particle rectangle. The summation of values in a rectangular region is done by accessing only four pixels. Comparing the simple summation of all pixels in rectangle, as known in computer vision area, integral image-based summation saves time exponentially.

Fig. 4 shows the resultant locations of tracked objects. Only using original particle filters with a proper likelihood function can track multiple targets robustly. The gray cloud is the distribution particles and green circle is the expected position of the particle distribution. This works fine until two objects are not near each other. Next section represents how to minimize the hijacking errors.

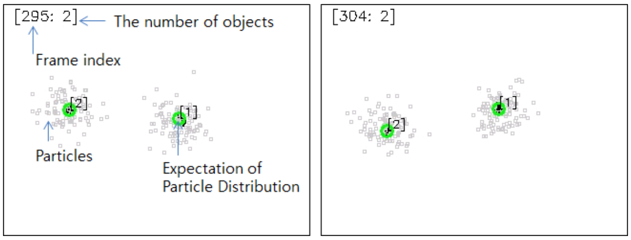


Fig. 4 Two frames (#295, #304) of tracking results in which the identities maintain consistently

3 MRF-Based Particle Filters

Independent particle filters are useful for single target tracking where no close and severe interaction happens between target objects. However when simultaneously tracking multiple objects, usually some of targets must get near or can cross over each other, then wrong target can hijack other tracking filters from neighboring object as shown in Fig. 5.

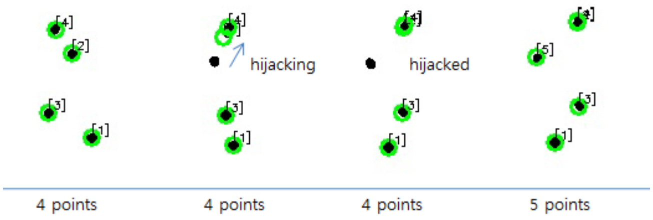


Fig. 5 As some objects get close their particle, Their weights can be wrongly augmented by other objects since the likelihood function has no guidance to the object identities

Hijacking other tracking filters on neighbors happens when some particles of the target are wrongly weighted by a near target. To avoid hijacking problem, Khan introduced a penalty function in the likelihood measurement using pairwise term of MRF motion model. The penalty function needs a graph for multiple objects. As a graph consists of vertexes and edges, in this multiple object tracking the graph has all gathered object positions as vertexes and connects them as links when the distance of objects is within the minimum edge distance as shown in Fig. 6. There can be several local graphs for the groups of locally neighbored objects.

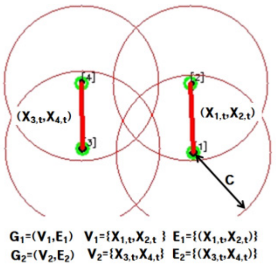

[4]
[3]
[2]
[1]
$(X_{3,t}, X_{4,t})$
$(X_{1,t}, X_{2,t})$
C
$G_1=(V_1,E_1)$ $V_1=\{X_{1,t},X_{2,t}\}$ $E_1=\{(X_{1,t},X_{2,t})\}$
$G_2=(V_2,E_2)$ $V_2=\{X_{3,t},X_{4,t}\}$ $E_2=\{(X_{3,t},X_{4,t})\}$

Fig. 6 Two graphs have been built based on the minimum edge distance *C*. Each graph has two vertexes and one edge.

The penalty function lowers the weights for those particles $x_{i,t}^{(s)}$which are very near neighboring target $x_{j,t}$. If particle is getting near the neighbor, penalty value for the particle's weight is getting higher to reduce the effect of this particle in the tracking filter. The penalty function of particle $x_{i,t}^{(s)}$ with neighbor $x_{j,t}$ is

$$\varphi\left(x_{i,t}^{(s)}, x_{j,t}\right) = \begin{cases} \dfrac{C - \sqrt{\left(x_{i,t}^{(s)} - x_{j,t}\right)^2}}{C} & if \sqrt{\left(x_{i,t}^{(s)} - x_{j,t}\right)^2} \le C \\ 0 & otherwise \end{cases}$$

where the minimum amount of penalty function is zero when the indexed particle is almost beside of a neighboring target. If the distance between the particle and neighbors is not less than C, penalty value is always one.

To apply the penalty effect, the particle is weighted with the likelihood function and the penalty function using

$$w_{i,t}^{(s)} = p\left(y_t | x_{i,t}^{(s)}\right)\left\{1 - \varphi\left(x_{i,t}^{(s)}, x_{j,t}\right)\right\}$$

Where x_{jt} is the neighbor which is in the minimum distance with the indexed particle. Fig. 7 shows the penalty effects on the particles and how different with Fig.5 to avoid the hijacking problem.

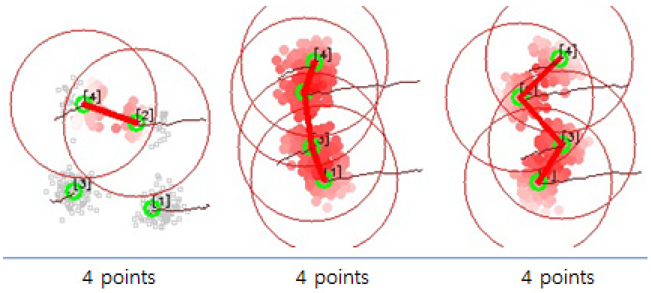


Fig. 7 The penalty effects are relatively colored as red on the locations of particles and by comparing Fig. 5 this approach can avoid the hijacking problem

However the time of the graph made is behind of one frame time. To eliminate the effect of one-frame late problem, we define a unary term to utilize the observation image. When the particles are positioned above neighbors, the likelihood function gives the maximum value. Therefore the previous penalty function needs to be changed to avoid the wrong maximum value of likelihood function. The unary term we define is used to check whether the particle rectangular area has white pixel or not and if it is near neighboring targets. The modified penalty function is

$$\varphi^*\left(x_{i,t}^{(s)}, x_{j,t}\right) = \varphi\left(x_{i,t}^{(s)}, x_{j,t}\right)h\left(x_{i,t}^{(s)}, x_{j,t}\right),$$

$$h\left(x_{i,t}^{(s)}, x_{j,t}\right) = \begin{cases} \left\{1 - u\left(\left(\grave{y}_t\left(x_{i,t}^{(s)}:1\right) - \grave{y}_t\left(x_{i,t}^{(s)}:2\right) - \grave{y}_t\left(x_{i,t}^{(s)}:3\right) + \grave{y}_t\left(x_{i,t}^{(s)}:4\right)\right)\right)\right\} & if \text{ case } A \\ 1 & otherwise \end{cases}$$

where h is the modifier for penalty function. The case A means that the indexed particle is near neighbor and not near its target. In case A, if particle overlaps the neighbor not its target, the value of the modifier h is zero. u is a step function. Fig. 8 shows the effects of the modifier.as blue color.

Based on the penalty function with unary and pairwise terms, it is possible to reduce the hijacking occurrences. The modifier function helped the proposed method effectively avoid hijacking problem.

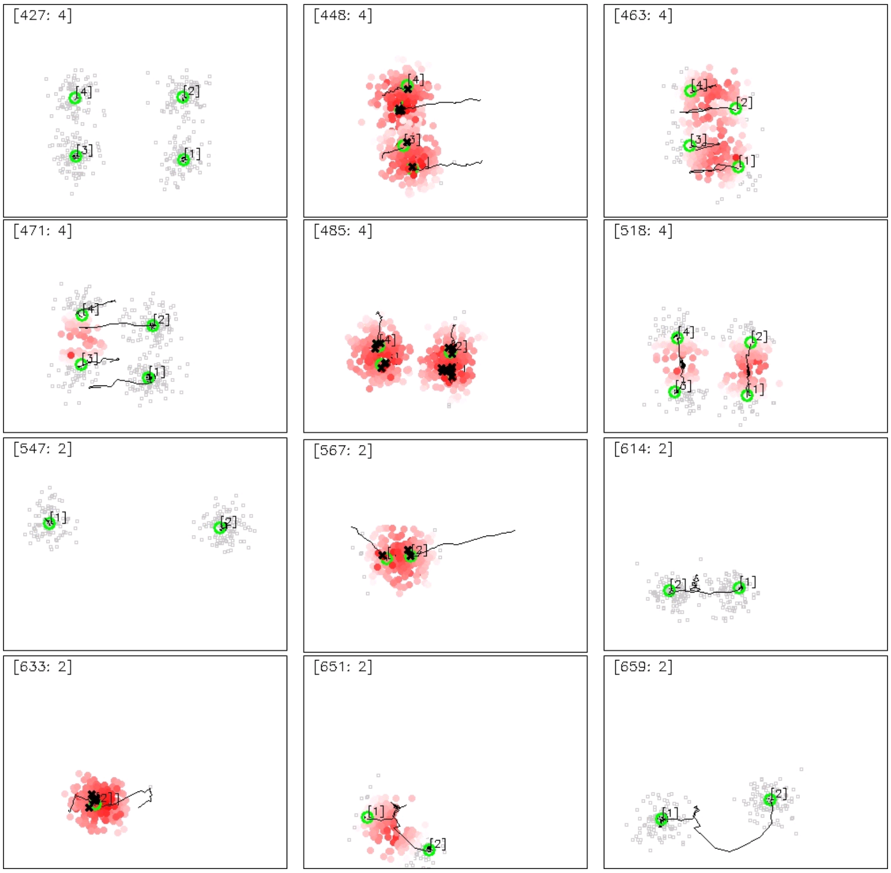


Fig. 8 The effect of the unary term-based modifier function shows that the particle with wrong maximum likelihood since the particle is located on a neighbor. And those particles are marked as black × and will have zero weight.

4 Action Likelihood Estimation

The trajectory of each graph shows the motion information of grouped objects such as translation, rotation and scaling with the description of the chord information of touch points. The chord information means how many points are moving or stable considering the structure of graph vertexes. The potential functions of the graph have the action information of multiple touch points. Therefore using the potential functions *(f1, f2, f3)* as shown in fig. 9 it is possible to establish action likelihood as

$$p_{transltion}\left(f_1|G_{i,t-1:t}\right) = \prod\nolimits_{(X_i,X_j)\in E}\left[\frac{(dx,dy)_i(dx,dy)_j^T}{|(dx,dy)_i||(dx,dy)_j\ |}\right],$$

$$p_{scaling}\left(f_2|G_{i,t-1:t}\right) = \prod\nolimits_{(X_i,X_j)\in E} u\left(\left|d_{i,j,t} - d_{i,j,t-1}\right| - S_{MIN}\right),$$

$$p_{rotation}\left(f_3 | G_{i,t-1:t}\right) = \prod_{(X_i, X_j) \in E} u\left(\left|\emptyset_{i,t} - \emptyset_{i,t-1}\right| - R_{MIN}\right)$$

where u is step function and the translation likelihood $p_{translation}$ is the normalized inner product between all displacement vector, the scaling likelihood $p_{scaling}$ means whether all points are scaled larger than S_{MIN}, and the rotation likelihood $p_{rotation}$ means whether all points are rotated larger than R_{MIN}.. These motion likelihoods can be measured concurrently.

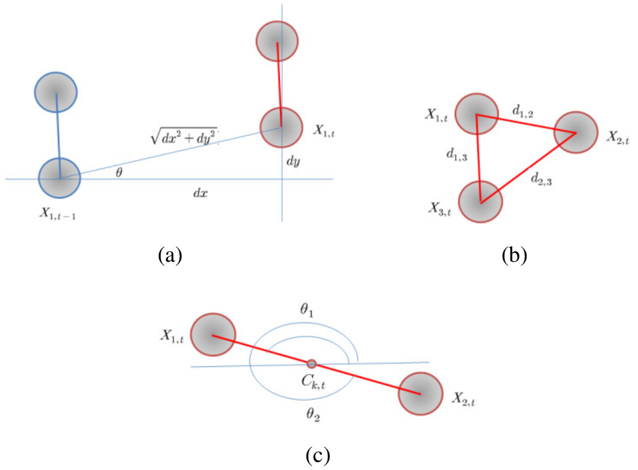


Fig. 9 the potential functions (f1, f2, f3) captures the motion information of vertex in two sequential time steps. (a) $f_1(X_{i,t}, X_{i,t-1}) = [dx, dy, \emptyset]^T$ (b) $f_2(X_{i,t}, X_{i,t-1}) = d_{i,j}$ (c) $f_3(X_{i,t}, C_{k,t}) = d_{i,j}$

Another potential function $f_4=[n, n_{move}, n_{stable}, n_{in}, n_{out}]^T$ represents the chord information of a graph. n is the number of touch points. n_{move} is the number of moving touch points, n_{stable} is the number of nonmoving points, n_{in} is the number of incoming points and n_{out} is the number of outgoing points between time t-1 and t. Considering the number of f_4 parameters five chord likelihoods can be defined by

$$p_{chord\_N}\left(f_4, N | G_{i,t-1:t}\right) = \int \delta(N - n),$$

$$p_{chord\_N_{move}}\left(f_4, N_{move} | G_{i,t-1:t}\right) = \int \delta(N_{move} - n_{move}),$$

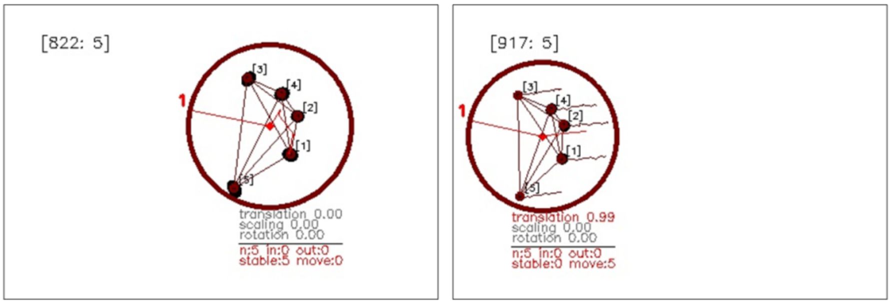
$$p_{chord\_N_{stable}}\left(f_4, N_{stable} | G_{i,t-1:t}\right) = \int \delta(N_{stable} - n_{stable}),$$

$$p_{chord\_N_{in}}\left(f_4, N_{in} | G_{i,t-1:t}\right) = \int \delta(N_{in} - n_{in}),$$

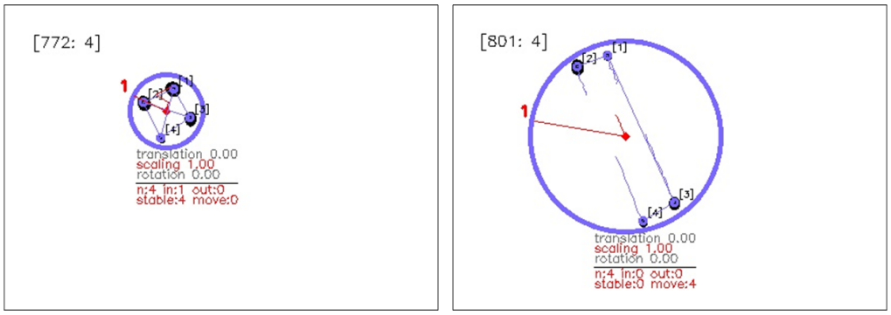
$$p_{chord\_N_{out}}\left(f_4, N_{out} | G_{i,t-1:t}\right) = \int \delta(N_{out} - n_{out}),$$

where N, N_{move}, N_{stable}, N_{in} and N_{out} are user queries on the graph, and δ is delta function; $\delta(0)$ is ∞ or otherwise zero.

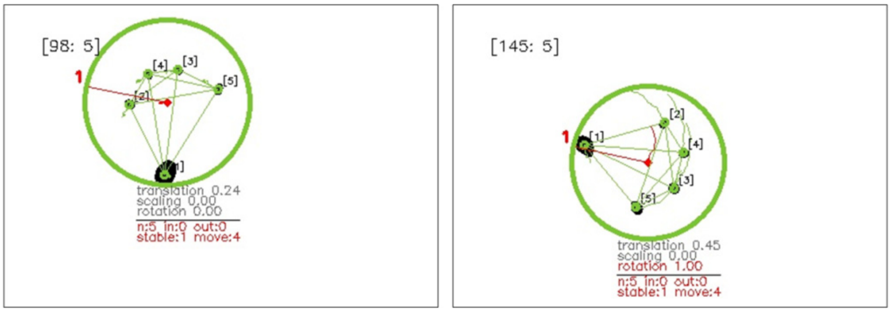
We have tested all action likelihoods and the estimated likelihoods and chord information in several motions as shown in Fig. 10. With all defined action likelihoods (motion, chord), any multi-touch gesture command in real-time application can be defined as a joint likelihood with them. For example, when we simply



(a) translation



(b) scaling



(c) rotation

Fig. 10 The action likelihoods of translation, scaling and rotation, and the chord information of $f_4=[n, n_{move}, n_{stable}, n_{in}, n_{out}]^T$ are shown below graph

use a multi-touch action (drag with two touch points) for mouse-drag command, the mouse-drag command can be defined as

$$p_{drag\_command} = p_{motion} p_{chord\_N}\left(f_4, 2 | G_{i,t-1:t}\right),$$

$$\text{where} \quad p_{motion} = p_{translation} p_{scaling} p_{rotation}$$

5 Conclusion

When we use independent particle filters for multiple object tracking, some target objects can lose their trackers due to the influence of the wrong maximum likelihood of neighboring target objects. This is called the hijacking problem and usually happens when target objects gathered into a small area. To avoid the hijacking problem the proposed method is MRF-Particle filters where MRF motion model is combined with particle filters to lower the weights of particles. If particles are very close to the neighbors then those weights are reduced by the penalty function of both pairwise and unary terms to avoid hijacking problem.

Due to the weakness of the pairwise term defined from the outdated graph, we additionally define a unary term to compensate the penalty function using given the observation under present time. The unary term captures the wrong maximum likelihood obtained from the neighbors. By reducing the effect of the wrong maximum likelihood, we can reduce the hijacking occurrences. This evaluation will be conducted in future works.

Additionally we expect that the information of MRF graphs can be used for action recognition. We have proposed how to define action likelihoods and joint likelihoods with potential functions which define some action elements and chord information. Some basic information of actions can be measured as shown in the resultant images. Future work will include extensive gesture recognition works based on these user-defined action likelihoods.

Acknowledgement. This research was supported by the MKE(The Ministry of Knowledge Economy), Korea, under the ITRC(Information Technology Research Center) support program supervised by the NIPA(National IT Industry Promotion Agency)" (NIPA-2013-H0301-12-3005).

References

1. Reid, D.: An Algorithm for Tracking Multiple Targets. IEEE Trans. on Automation and Control AC 24, 84–90 (1979)
2. Bar-Shalom, Y., Fortmann, T., Scheffe, M.: Joint Probabilistic Data Association for Multiple Targets in Clutter. In: Proc. Conf. on Information Sciences and Systems (1980)
3. Schulz, D., Burgard, W., Fox, D., Cremers, A.B.: Tracking Multiple Moving Targets with a Mobile Robot using Particle Filters and Statistical Data Association. In: IEEE Int. Conf. on Robotics and Automation (2001)

4. Khan, Z., Balch, T., Dellaert, F.: MCMC-based Particle Filtering for Tracking a Variable Number of Interacting Targets. PAMI 27(11), 1805–1819 (2005)
5. Viola, P., Jones, M.: Robust Real-time Object Detection. Int. Journal of Computer Vision 57(2), 137–154 (2002)
6. Islam, M.Z., Oh, C.M., Lee, J.S., Lee, C.W.: Multi-Part Histogram based Visual Tracking with Maximum of Posteriori. In: Int. Conf. on Computer Engineering and Technology (2010)
7. Stalder, S., Grabner, H., Van Gool, L.: Cascaded confidence filtering for improved tracking-by-detection. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part I. LNCS, vol. 6311, pp. 369–382. Springer, Heidelberg (2010)
8. Han, J.Y.: Low-cost multi-touch sensing through frustrated total internal reflection. In: Proc. 18th Annu. Assoc. Comput. Machinery Symp. User Interface Software Technology, pp. 115–118 (2005)
9. Oh, C.M., Lee, Y.C., Lee, C.W.: MRF-based Particle Filters for Multi-touch Tracking and Gesture Likelihoods. In: 11th IEEE International Conference on Computer and Information Technology, pp. 144–149 (2011)