2D Hand Tracking Based on Flocking with Obstacle Avoidance

Regular Paper

Zihong Chen¹, Lingxiang Zheng¹,*, Yuqi Chen¹ and Yixiong Zhang¹

¹ School of Information Science and Engineering, Xiamen University, Xiamen, China
* Corresponding author E-mail: lxzheng@xmu.edu.com

Received 08 Jan 2013; Accepted 26 Nov 2013
DOI: 10.5772/57450

© 2014 The Author(s). Licensee InTech. This is an open access article distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/3.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Abstract Hand gesture-based interaction provides a natural and powerful means for human-computer interaction. It is also a good interface for human-robot interaction. However, most of the existing proposals are likely to fail when they meet some skin-coloured objects, especially the face region. In this paper, we present a novel hand tracking method which can track the features of the hand based on the obstacle avoidance flocking behaviour model to overcome skin-coloured distractions. It allows features to be split into two groups under severe distractions and merge later. The experiment results show that our method can track the hand in a cluttered background or when passing the face, while the Flocking of Features (FoF) and the Mean Shift Embedded Particle Filter (MSEPF) methods may fail. These results suggest that our method has better performance in comparison with the previous methods. It may therefore be helpful to promote the use of the hand gesture-based human-robot interaction method.

Keywords Human-Computer Interaction, Human-Robot Interaction, Hand Tracking, Flocking Behaviour, Obstacle Avoidance

1. Introduction

As various computer applications penetrate into our daily life, people are in great need of natural and efficient human-computer interaction (HCI). It is likely that in the future many service robots with different sets of skills will interact with humans [1] [2]. Researchers have made efforts to develop more natural HCI, such as speech, posture and gesture recognition for controlling robotic applications [3]. Gesture-based interfaces provide an intuitive way for users to specify commands and interact with computers. Based on the movement of the hand, the robot is moved under control in the respective direction, i.e., forward, backward, left or right [4] [5] [6].

Hand gesture, as a natural and expressive interaction modality, has gained much more attention. The existing hand gesture interface techniques can be categorized into glove-based and vision-based. Compared with expensive and cumbersome glove-based devices, the touch-free interface would be preferred by most people. Furthermore, with the advances in the fields of computer vision, image processing and pattern recognition, vision-based applications have become feasible and could develop to replace the glove devices.

The existing vision-based hand recognition systems can be classified into 3D model-based [7] [8] [9] [10] and 2D appearance-based [11] [12]. The 3D-based method has many advantages. However, a depth camera is much more expensive than a traditional web camera which limits the wide-spread use of the depth camera in gesture recognition systems. Therefore, considering its economy, simplicity and practical feasibility, we primarily focus on the 2D image plane which is based on the cheap
web camera. A 2D vision-based hand gesture recognition system contains three phases: hand detection, hand tracking and hand recognition. Hand tracking is a pivotal phase in a hand gesture recognition system. In this phase, a hand tracker tracks the hand’s movements and records the hand trajectory for further processing. In the recognition phase, the system recognizes the meaning of the trajectory represented which can be translated to a HCI control command for communication.

There has been substantial research into the area of hand tracking. H. Fei and I. Reid [11] developed a tracking system which combines the Particle Filter (PF) and the Hidden Markov Model (HMM) for the tracking and recognition of articulated hand motion. The PF is adopted for colour-region tracking to assist the HMM in analysing hand shape variations. In turn, the HMM filter outputs the shape and provides the important weights of the particles. However, this method assumes that the hand moves with cyclic shape variations. Users cannot move naturally and the tracker should be tailored for different application systems according to predefined shapes. In [13], the authors proposed a tracking algorithm, the Mean Shift Embedded Particle Filter (MSEPF), to implement a wheelchair control interface. In MSEPF, mean shift is performed on each particle to move it toward local maximum after they are propagated. It can reduce the number of particles and thus lower the computational cost. The tracker is robust to rapid movement and various kinds of distractions. However, it would fail when the hand moves over the face or other big skin-coloured objects slowly. In [14], the authors proposed a hand tracking system that accommodates multiple hypotheses for the hand location in each frame. A set of rules is applied to select the best candidate. The method can afford to use a relatively simple and efficient method for feature extraction and estimation. However, considering the computation cost, it needs to train pruning classifiers from training samples to eliminate unlikely hypotheses. Every time we change the set of gestures, we need to train the tracker’s pruning classifiers.

M. Kölsch and M. Turk proposed the Flocks of Features (FoF) algorithm [15] without any of the drawbacks mentioned above. It can track human hand continuously despite rapid hand movements and pose variations. It combines the KLT (Kanade-Lucas-Tomasi) feature, flocking behaviour, and a learned foreground colour distribution. KLT is an implementation of a feature tracker for the computer vision community. Based on the early work of B. D. Lucas and T. Kanade [16], KLT was fully developed by C. Tomasi and T. Kanade [17] and was explained clearly in the paper by J. Shi and C. Tomasi [18]. The speed of the pyramid-based KLT feature tracking [18] allows the method to achieve the real-time performance easily. Motivated by the seemingly chaotic clustering behaviour of a school of fish or a flock of birds such as pigeons [19], it was introduced to make possible the tracking of objects whose appearances change over time by enforcing some constrains on the feature locations. It can only describe the collective behaviour without obstacles. However, the FoF assumes that no other exposed skin area is within the background reference area, so it will fail when the hand moves in front of some skin-coloured objects, especially the face.

Researchers made some improvements based on the FoF. In [20], M. Kölsch introduced a new version of FoF which adds appearance as an additional image cue. This could help to overcome tracking difficulties in texture-rich and skin-coloured environments, but one more cue requires more computations. Its computation time rises from 2-18ms per 720x480 RGB frame [15] to around 20ms [20]. A. Fogelton proposed an improvement to the FoF by processing the frame using histogram back projection of the skin colour prior to applying flocks of features. This modification provides better results with a lower false positive error rate when hand movements are over the strong edges, e.g., the edges of picture frames hanging on the wall [21]. However, the failure due to skin-coloured objects distraction is not tackled in those improvements to the FoF.

The main contribution of this paper is to provide a hand tracking method based on the Flocking Model with Obstacle Avoidance (FMOA) to overcome the failure which is caused by distraction meanwhile keeping good performance. When tracking the hand under severe distraction, the flocking behaviour allows the flock of features to split into two candidates for tracking. The hand will be distinguished from the distracters several frames later and the two flocks merge into one again. Our method allows the hand to move more freely and also allows for more precise tracking, which spares us the need to worry about the skin-coloured distraction in real life scenarios, e.g., scenarios when an individual is controlling a service robot to move left or right to avoid obstacles, or when an individual is sitting in front of a television set, a web camera or a laptop with a built-in camera.

The organization of the rest of this paper is as follows. Section 2 presents the hand tracking method that uses flocking with obstacle avoidance. Section 3 describes the test data and evaluation method. In section 4, we present and discuss the results. Finally we draw some conclusions.

2. Method

When tracking the hand, our tracker integrates two image cues in a very natural manner. The KLT feature is the first cue that delivers excellent results on rapid moving rigid objects and can be computed very efficiently [18] [22]. The colour as the second cue is consulted to aid in the choice of location when initializing or relocating features. Before the tracker starts, the hand region is located by the initial KLT features which are selected according to the combined probability of their locations and colour. Then, the flock of features is controlled by flocking behaviour and each KLT feature is tracked individually from frame to frame. In order to overcome some big skin-coloured distractions, our tracker works with the obstacle avoidance flocking behaviour, and the flock splits into two sub-flocks to pass the static obstacles in a cluttered background.

2.1. Flocking Model with Obstacle Avoidance

Flocking is a form of collective behaviour exhibited when a group of agents, called a flock, move together and interact with one another. In 1987, C.W. Reynolds proposed a classic flocking model that led to the creation of the first computer animation of flocking [23]. It consists of three heuristic flocking rules: Collision Avoidance (CA), Flock Centring (FC), and Velocity Matching (VM).

The flocking model can only describe the collective behaviour without obstacles. Many other rules have been added to the flocking algorithm, such as obstacle avoidance. H.G. Tanner [24] and R. Olfati-Saber [25] extended the flocking model with obstacle avoidance (FMOA) to avoid several static obstacles in the environment. The flocking model is described as follows. The adjacency undirected graph $G(V, E)$ is used to describe the N agents of the flocking model with obstacle avoidance, where $V = \{1, 2, ..., N\}$ is a set of vertices (nodes), $E = \{(i, j) \in V \times V \}$ is a set of edges that represent the neighbouring relations among the N agents and $N_i$ is a set of agents that is neighboring to agent $i$,

$$ N_i = \{ j \in V \mid d(i, j) \leq \xi_{nby} \} $$

where $d(i, j)$ is the relative distance between agent $i$ and $j$, $\xi_{nby}$ defines the maximum distance between two neighbourhood features. The relative distance of features in the moving flock is constantly changing, which also affects the adjacency relation.

The flocking group of N agents, moving on the plane with the following dynamics:

$$ \begin{align*}
    p_i &= v_i \\
    v_i &= u_i \\
    i &= 1, 2, ... N
\end{align*} $$

where $p_i = (x_i, y_i)$ represents the location of agent $i$ on the plane, $v_i = (\dot{x}_i, \dot{y}_i)$ defines the velocity of agent $i$ and $u_i = (u_{x_i}, u_{y_i})$ is the control inputs which consist of three components:

$$ u_i = a_i + \beta_i + \chi_i $$

The first component $a_i$ is generated by an artificial potential field that controls the cohesion and separation of the agents within the group. The second component $\beta_i$ regulates the velocity vector of agent $i$ to the average of its own and those of its neighbours. The last component $\chi_i$ is a second artificial potential field that ensures the agents avoid stationary obstacles and move towards a predefined position.

2.2. Hand Tracking using Flocking with Obstacle Avoidance

In the KLT feature-based hand tracking (FoF), C.W. Reynolds’s flocking rules (CA and FC) have been adopted to enforce a loose global constraint on the feature locations that keeps them spatially confined [15]. In addition the flock features will be stopped by some big skin-coloured obstacles, especially the face region. Thus, we apply the FMOA to our method (HTFOA), for overcoming the distraction.

Consider the KLT feature set $V = \{1, 2, ... N\}$, and $p_i(t) = (x_i, y_i)$ represents the position of feature $i$ at the time $t$. Let $N_i$ denote the neighbor sets of feature $i$. Let $N_{nby}$ denote the smallest number of features in $N_i$. Let $N'_i$ denote the set of feature $i$ which has $N_{nby}$ number of nearest features.

$$ N_i = \{ j \in V \mid d(i, j) \leq \xi_{nby} \} \cup N'_i $$

The definition ensures that any feature $i$ always has at least $N_{nby}$ number of features around it and also is a reference when the features split into two sub-flocks. The nearby parameter $\xi_{nby}$ defines the maximum distance between two features that each can be the other’s nearby flock mate.

The flock centre is indicated by Median Feature (MF),

$$ MF = \text{argmin}_{i \in V} \xi_{j \in V} d(i, j) $$

where MF is the feature in the set $V = \{1, 2, ..., N\}$ that has the minimum distance $d(i, j)$ between features $i$ and $j$.

The median feature of $i$’s neighbour set $N_i$ is $MF_{nby}(i)$,

$$ MF_{nby}(i) = \text{argmin}_{j \in N_i \cap N_k} d(j, k) $$

From the time $t$ to $t+1$, feature $i$’s movement vector $d_i$ is:

$$ d_i = d^{dt}_i + d^{sa}_i + d^{fc}_i $$

where $d^{dt}_i$ is the displacement vector that uses the KLT algorithm to update the position of feature $i$ when there is no distraction. Feature $i$ will be stopped by distraction and $d^{dt}_i$ will be assigned the value zero.

$d^{sa}_i$ is the displacement vector caused by the Collision Avoidance (CA) rule to adjust feature $i$’s position by randomly placing on the flock region according to the skin probability, and it occurs if:

$$ \begin{align*}
    \exists f_j &\in V \\
    d(i, j) &< \xi_{col} \quad j \neq i
\end{align*} $$

where collision parameter $\xi_{col}$ indicates the distance between two features when a collision occurs.

$d^{fc}_i$ is the displacement vector caused by the rule Flock Centring (FC) to prevent feature $i$ from straying from the MF, it occurs if:

$$ \begin{align*}
    d(i, MF) &> \xi_{cen} \\
    d(i, MF_{nby}(i)) &> \xi_{nby}
\end{align*} $$

where the centring parameter $\xi_{cen}$ defines the maximum distance between feature $i$ and the MF. Feature $i$ has strayed when it is far away from the MF and $MF_{nby}(i)$.

The stray feature $i$ will be moved by:

$$ p_i = p_i + (MF_{nby}(i) - p_i)/2 $$

until $d(i, MF_{nby}(i)) \leq \xi_{nby}$.

If the flocking features meet the obstacles at time $t$, the distance will be larger than split parameter $\xi_{spl}$:

$$ d(i, MF_{nby}(i)) > \xi_{spl} $$
Some features will be stopped by obstacles, then the flock will automatically split into two sub-flocks, called flock a and flock b:

\[
\begin{aligned}
V_a &= N \\
V_b &= V - V_a
\end{aligned}
\]  

(12)

The two sub-flocks a and b will be tracked by the tracker independently (just like the FoF tracker). Consider \(MF_a\) to be the median feature of the sub-flock a. The stray feature \(i\) will be dragged into the flock region by \(p_i = p_i + (MF_a - p_i)/2\) until \(i\)'s location satisfies the relation: \(d(i, MF_a) \leq \xi_{cen}\).

The parameter \(\xi_t\) is the time interval for the tracker to calculate the moving distance:

\[
\begin{align*}
move\_range_1 &= d(p_{MF}(t), p_{MF}(t + \xi/2)) \\
move\_range_2 &= d(p_{MF}(t + \xi/2), p_{MF}(t + \xi)) \\
move\_range_a &= move\_range_1 + move\_range_2
\end{align*}
\]

(13)

and judge which sub-flock is dragged by the obstacles and merge this sub-flock into the other one which avoids obstacles. If the sub-flock moves back to the original position from \(t\) to \(t + \xi_t\), then its moving distance is zero, which may be regarded as obstacles.

Hence, we calculate each sub-flock’s moving distance in two time intervals \([t, t + \xi_t/2]\) and \([t + \xi_t/2, t + \xi_t]\).

3. Experiments

We test the performance of our HTFOA on 42 video sequences\(^1\) in comparison with the FoF and the MSEPF. The videos were captured by a USB webcam at a resolution of 640×480 pixels and a rate of 30 frames per second. Four people (male and female) were invited to record their hand motions at different times of the day in an office environment. Table 1 details the sequences’ main characteristics. The 42 video sequences contain possible tracking failures or skin-coloured obstacles, in which case the common tracking methods may fail when they move over the face region, rapid hand movements, changing postures, luminance variation and so on.

The MSEPF tracker is configured with 100 particles to track. The FoF tracker is an integral part of HandVu [26], the first vision-based hand gesture interface that is publicly available. While ours is implemented based on HandVu using the default configuration with \(N_f = 50\), \(\xi_{cen} = 3.0\) and \(\xi_{nby}\) and \(\xi_{spl}\) are fixed as \(\xi_{cen} = \xi_{nby} = (\text{detected hand region width + height})/4\), \(\xi_{spl} = (\text{detected hand region width + height})/2\). When the flock splits, the two sub-flocks were tracked by the tracker and their moving distances calculated the moving distances after the time interval \(\xi_t\) of 10 frames independently, the two sub-flocks will be distinguished according to their moving distances, and the dragged one would be merged to the other one. We experimented with different values for parameter \(N_{nby}\) while further investigations for \(\xi_{nby}\) and \(\xi_{spl}\) are necessary.

4. Results and Discussion

In our experiments, the hand is the tracking target. The exposed skin area of the wrist and arm is not to be considered as the part of the hand. The track target is considered lost when the mean location (the centroid of the detected hand\(^2\)) is no longer on the hand. The trajectory starts from the moment the tracker detects the hand and ends when the tracker loses the target. Noncontinuous trajectory may cause ambiguity in the information for hand recognition. Thus, even if the tracker can recover the tracking after it loses the target, the trajectory is not continuous, which makes it difficult to use in hand recognition. Therefore, we only focus on the trajectory before the tracker loses the target.

4.1. Compared with FoF and MSEPF

Our method tracked the hand by tracking the flock of the KLT features, and according to our flocking model (FMFAO), the flock splits into two sub-flocks when it meets the face region or other distracters, and merges later. Figure 1 depicts how our HTFOA avoids the face obstacles during moving, and the comparison with the other two methods are also shown. In Figure 1, the cyclic hand motion moves from the right side to the left side of the face area and then returns to the initial position.

Our tracker recorded the hand trajectory over a longer period of time than the other two methods as illustrated in Figure 1. The hand moved from the right side to the left side over the face region from Frame 125 to Frame 158, then returned to the right side from Frame 158 to Frame 195. In our method, the flock met the face region, split into two targets on Frame 156 and later merged on Frame 173, then split on Frame 185 and merged on Frame 195 later, while the MSEPF failed on Frame 156 and the FoF on Frame 181, the centroid of the tracking window is wrongly located on the face region.

When the hand moved close to where it is about to be lost from Frame 158 to Frame 173, the MSEPF caught the hand coincidentally on Frame 173. Such a situation will produce two hand trajectories. Just as we have mentioned, two hand trajectories may cause ambiguity. Therefore we treat it as a failure in the tracking phase.

---

\(^1\) The videos are available upon request.

\(^2\) In MSEPF, the centroid of the detected hand represents the centroid of the tracking window. In FoF and HTFOA, it represents the centroid of the flock of features.

<table>
<thead>
<tr>
<th>Type Id</th>
<th>Movement Speed</th>
<th>Posture Changes</th>
<th>Sleeves</th>
<th>Sequences Number</th>
<th>Len Frames</th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td>Slow</td>
<td>No</td>
<td>Long</td>
<td>7</td>
<td>4959</td>
</tr>
<tr>
<td>001</td>
<td>Slow</td>
<td>No</td>
<td>Short</td>
<td>6</td>
<td>4756</td>
</tr>
<tr>
<td>010</td>
<td>Slow</td>
<td>Yes</td>
<td>Long</td>
<td>4</td>
<td>2862</td>
</tr>
<tr>
<td>011</td>
<td>Slow</td>
<td>Yes</td>
<td>Short</td>
<td>6</td>
<td>4066</td>
</tr>
<tr>
<td>100</td>
<td>Quick</td>
<td>No</td>
<td>Long</td>
<td>5</td>
<td>1933</td>
</tr>
<tr>
<td>101</td>
<td>Quick</td>
<td>No</td>
<td>Short</td>
<td>4</td>
<td>1692</td>
</tr>
<tr>
<td>110</td>
<td>Quick</td>
<td>Yes</td>
<td>Long</td>
<td>2</td>
<td>582</td>
</tr>
<tr>
<td>111</td>
<td>Quick</td>
<td>Yes</td>
<td>Short</td>
<td>8</td>
<td>3314</td>
</tr>
</tbody>
</table>

Table 1. The video sequences. Len frames is the number of video frames.
Figure 1. Snapshots from the tracking result video using the MSEPF, the FoF and the HTFOA; the big dot is the centroid of the detected hand and the yellow titles show their frame numbers: 125, 137, 149, 156, 158, 173, 175, 181, 185 and 195

Figure 2. Snapshots from the tracking result video using the MSEPF, the FoF and the HTFOA in a more cluttered background; the big dot is the centroid of the detected hand and the yellow titles show their frame numbers: 308, 329, 333, 341, 342, 420, 423, 428, 433 and 435

We record the frame length from the first time the hand is tracked to the time the hand is lost, the frame length in all eight situations are depicted in Table 2. The tracking frame lengths that are listed in Table 2 do not contain the hand detection phase.

<table>
<thead>
<tr>
<th>Video type id</th>
<th>Tracking method</th>
<th>MSEPF</th>
<th>FoF</th>
<th>HTFOA</th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td></td>
<td>981</td>
<td>1898</td>
<td>3074</td>
</tr>
<tr>
<td>001</td>
<td></td>
<td>211</td>
<td>2607</td>
<td>3346</td>
</tr>
<tr>
<td>010</td>
<td></td>
<td>872</td>
<td>1453</td>
<td>1641</td>
</tr>
<tr>
<td>011</td>
<td></td>
<td>295</td>
<td>1552</td>
<td>1945</td>
</tr>
<tr>
<td>100</td>
<td></td>
<td>164</td>
<td>741</td>
<td>895</td>
</tr>
<tr>
<td>101</td>
<td></td>
<td>78</td>
<td>597</td>
<td>624</td>
</tr>
<tr>
<td>110</td>
<td></td>
<td>17</td>
<td>252</td>
<td>286</td>
</tr>
<tr>
<td>111</td>
<td></td>
<td>100</td>
<td>1248</td>
<td>1404</td>
</tr>
</tbody>
</table>

Table 2. The length of the tracking frames. The video types are already depicted in Table 1.

As expected, our method performs better in all test cases than the FoF and the MSEPF. When the hand is moving at a low speed (as in type 000, 001, 010 and 011), all these methods track well, and the frame length depends on when the hand meets the face region obstacle, while in rapid movement (as in type 100, 101, 110 and 111), all methods will easily lose the target before the hand meets the face. The MSEPF fails much more than the other two methods when the video contains skin-coloured obstacles, such as the face and arm regions. There is no significant difference in speed between the FoF tracker and our tracker, they both require around 12ms per 640x480 video frame. The speed of the MSEPF tracker per frame is about 1.372ms per particle, which means 100 particles need around 138ms in total.

Figure 1 and Figure 2 both illustrate our modified tracker’s performance in comparison with the FoF and the MSEPF in video type 001. Video type id 000-111 are eight kinds of video sequences with different gesture styles illustrated in Table 1. Figure 2 depicts a much more cluttered background that contains other people, and the
hand is moving around the face region. The MSEPf failed on Frame 433 when its centroid of the tracking window is located on the arm region and the FoF lost the hand when moving around the face on Frame 420.

4.2. Parameter Optimizations

The parameter of the least number of nearby flock mates \((N_{nby})\) influences a feature's nearby flock mates set. It has a great impact on the decisions about whether a feature of the flock strays and needs to be relocated or the flock needs to be split. A small value \(N_{nby}\) will result in frequent splits of the flock, while a large value may cause the failure of the tracker in avoiding obstacles. Figure 3 presents the tracking results after varying \(N_{nby}\). The tracker with \(N_{nby} = 20\) performs the best. The heights of the columns are calculated by the following equation,

\[
\begin{align*}
\text{sum(type\_id)} &= \text{sum(tracking lengths)} \\
\text{height} &= \text{sum(type\_id)} / \max\{\text{sum(type\_id)}\} \\
\text{type\_id} &= 000, 001, ..., 111
\end{align*}
\]

The \text{type\_id} is depicted in Table 1, \text{sum(type\_id)} is the sum of the tracking lengths in each video type, \(\max\{\text{sum(type\_id)}\}\) is the maximum in the eight \text{sum(type\_id)}.

Figure 4 (for female) and 5 (for male) show the results for our HTFOA with different values for \(N_{nby}\). The cloud of little dots represents the flock of features, and the big dot is their centroid.

If a relatively small value is given to \(N_{nby}\), such as 5, the tracker will be very sensitive to small distraction and the flock will be split frequently. This kind of small distraction can be handled by the centring behaviour without splitting the flock to track two candidate targets. Moreover, frequent splitting is more likely to result in tracking failure in a cluttered background (Figure 4(a)).

A relatively large \(N_{nby}\), such as 25, will perform more stably than small \(N_{nby}\) when processing small distractions. But when the hand encounters big skin-coloured objects like the face region, usually there are just a few features left on the hand. In that case, the tracker may not be aware of the severe distraction and will not switch into the status of tracking two targets. It will fail just like the FoF behaviours (Figure 5(e)). Their processing times are all around 12ms per frame when varying this parameter, without significant difference.

5. Conclusions

In this paper, a novel hand tracking method which applies the flocking model with obstacle avoidance is presented. We use the obstacle avoidance flocking behaviour for tracking the hand to avoid static obstacles. When there are severe background distractions, the flocking behaviour will make the tracking target divide into two parts, which are tracked by two separated flocks of features and merged after overcoming the skin-coloured distractions. Our method can handle a slow moving hand, and vast and rapid deformations well, and avoid skin-coloured obstacles. However, it may fail due to rapid movements, the lumiance variation of the skin, or in a situation that contains more than one moving hand, where the other two methods will also fail.

Generally speaking, our method performs better than the other two methods. The experiment results show that our method can record a longer trajectory. This implies that the HTFOA has better performance than the MSEPf and FoF when handling big skin-coloured object distractions. Moreover, in comparison with the FoF, it only requires 12ms per frame which only fractionally increases the computational cost. In the future, we are planning to tackle this problem by distinguishing the distractions in a much easier way and splitting into more sub-flocks when more cluttered backgrounds are presented. The improved tracking method will be integrated with hand gesture recognition to produce a real-time HCI system for future service robots.

6. Acknowledgement

This project is supported by the National Natural Science Foundation of China (no. 61102135), the Fundamental Research Funds for the Central Universities, P.R. China (no. 2010121063), and a grant from the Key Programmes of Science and Technique Commission Foundation of Fujian Province, P.R. China (no. 2011H6028).

**Figure 3.** How \(N_{nby}\) varying influences the performance for each of the video sequences that are illustrated in Table 1

**Figure 4.** Results for HTFOA with different values for \(N_{nby}\). The cloud of little dots represents the flock of features, the big dot is their centroid.

**Figure 5.** Results for HTFOA with different values for \(N_{nby}\). The cloud of little dots represents the flock of features, the big dot is their centroid.
7. References


