# Initialization of Multiple Objects Tracking using Flocking Behavior of KLT Features

Andrej FOGELTON[*]

*Slovak University of Technology*
*Faculty of Informatics and Information Technologies*
*Ilkovičova 3, 842 16 Bratislava, Slovakia*
fogelton@gmail.com

**Abstract.** Computer vision is closely linked to machine learning, where there is mostly a requirement for efficiency and real-time performance. As an example we present our modification of the Flocks of features algorithm, which is basically an efficient partial clustering method. The main contribution of this paper are two initialization methods used along this algorithm. The first static method is used to obtain multiple clusters with density based approach, which will be subsequently tracked by Flocks of features. The second dynamic method is used to detect new objects to track along already running FoF tracking on a frame sequence.

## 1 Introduction

Flocks of features (FoF) is a hand tracking algorithm with some clustering principles. This algorithm is background invariant, working without any gloves or other markers, light invariant, hand shape invariant and mostly able to track both human hands. Our data are represented by KLT (Kanade, Lucas, Tomasi) [5, 8] features, which can be efficiently tracked from one frame to another of the same sequence. We locate these features onto human hand to track them in a frame sequence.

There is a big difference between hand tracking and hand detection. FoF tracking needs clusters of points, which can be tracked over time. If we want to use hand tracking as an input device to interact with computer, we need to set up an automatic initialization. This paper proposes two initialization methods for FoF tracking; static and dynamic. There is a large number of clustering algorithms, which could be used for static initialization, but only few of them are eligible to cooperate with computer vision algorithms. We did research on clustering algorithms with specific demands because of our data representation and efficiency.

To understand the advantages and disadvantages of these initialization methods, we divided this paper into several sections. In section 2 we present flocks of features and our improvement to this algorithm. Section 3 gives a brief description of different clustering methods. Section 4 and 5 describes our initialization methods for the modification of FoF algorithm and in the last section we do conclusion and discuss possible future work.

[*] Master study programme in field: Software Engineering
Supervisor: Dr. Matej Makula, Institute of Applied Informatics, Faculty of Informatics and Information Technologies STU in Bratislava

*IIT.SRC 2011, Bratislava, May 4, 2011, pp. 1–6.*

## 2 Flocks of features and our modification

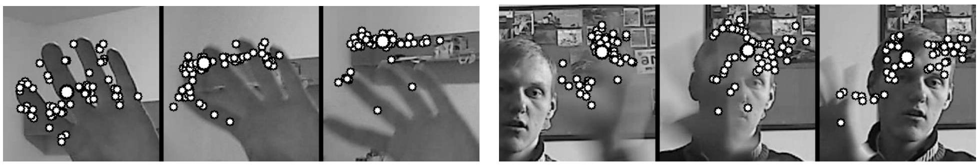
Mathias Kölsch and Matthew Turk presented a *Fast 2D hand tracking with flocks of features and multi-cue integration* [3]. This algorithm can track the human hand without any artificial objects such as gloves. It can be used in various light conditions and furthermore a non stationary camera can be used. The tracker's core idea is motivated by the seemingly chaotic flight behavior of a flock of birds [4] such as pigeons, where the minimum and maximum safe distance during the flight are defined. Features of the hand are also very close together like birds in a cloud [4].

The probability mask states for every pixel in the bounding box the likelihood that belongs to the hand. Features are selected within the bounding box according to their ranking and observing a pair wise minimum distance. These features are being ranked according to the combined probability of their locations and color. Highly ranked features are tracked individually per frames. Individual features can latch onto arbitrary artifacts of the object being tracked, such as fingers of a hand. Their movement is independent along with the artifact, without disturbing other features. Too dense concentrations of the features that would ignore other object's parts are avoided due to the minimum distance constraint. But stray features that are too far from the object of interest are brought back into the flock with the maximum distance constraint. To get more stable results, about 15% of the furthest features from median computation have to be removed. The speed of pyramid-based KLT feature tracking allows to overcome the computational limitations of tracking the model-based approaches and achieving real-time performance.

During calibration process, a hand color is observed and the normalized-RGB histogram is created. The color information is used as a probability map. At tracker initialization time, the KLT features are placed preferably onto locations with high skin color probability. New location of a relocated feature is chosen with high color probability (more than 50%). Changing light condition can cause bad tracking performance, but only in case of relocated features because most of the features will continue to follow gray level artifacts. This method combines cues from feature movement based on gray level image texture with cues from texture-less skin color probability. It depends on the algorithm parameters how often features are relocated and on the importance of the color modality.

### 2.1 Modification

The original FoF uses gray level image for KLT features tracking. Due to this procedure we found the FoF algorithm to be vulnerable to edges (Figure 1a) occurring in the background [1]. During movements over strong edges, many KLT features are tracked to incorrect positions. This leads to an incorrect median relocation and tracking failure (Figure 1). One slight difference is using the HSV color model instead of normalized-RGB, because it is more common in these kind of application and better while using histogram.



(a) (b)

Figure 1: In (a) Original FoF fails, but the modification does not (strong edges in the background) and in (b) both FoF fails (very fast hand movement over head, 30fps is not enough in these kind of situations).

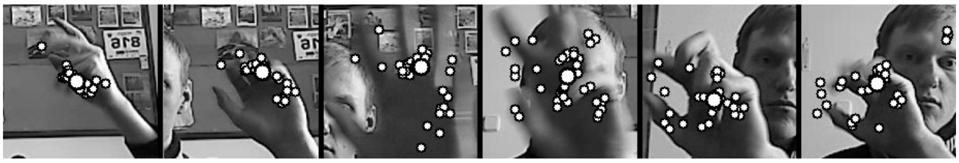


Figure 2: Continues tracking of various hand movements with pose changing.

We realize that if we use the probability map instead of the gray level image, we will get rid off edges in the background because they are not skin colored and they did not appear in the back projected image. The idea to run FoF on this probability map (Figure 3) has been proved to be a step forward. Because of this modification we do not need to rank features in the initialization procedure and we can be almost sure that every feature will be located somewhere in the skin region. This modification is more precisely described with testing results in paper [1].

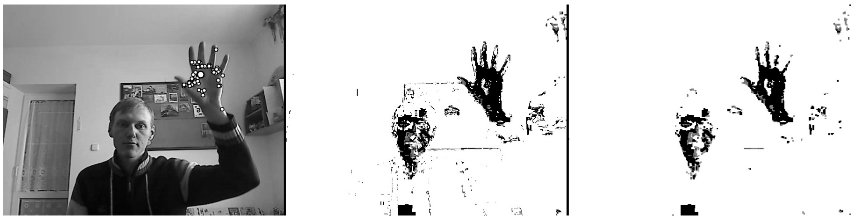


Figure 3: Original image with flock of features, result of histogram back projection with noise, removing noise with operation open (erosion and dilatation) [1].

## 3 Related work

In the back projected image, there is always some noise and on this noise KLT features can be also detected. This means that we need an algorithm which is robust to the outliers. Clusters are characterized by increased density against the background. From these assumptions we are looking for partial and density based algorithm, which will do unsupervised learning, because we do not know the exact number of clusters. It is an assumption that it can be from 0 to 3, because there can be just the user's head alone or with one hand or both hands. The other requirement is the performance of the algorithm, because we want to use our hand tracking technique as an input device to allow the user to interact with computer in a more natural way using hand gestures.

The most common algorithm for clustering is K-means [7], but this algorithm has two main disadvantages. It is supervised, we need to set the number of cluster we are looking for and it is not robust to outliers and that is why it is not appropriate for our problem.

### 3.1 Hierarchical clustering

Agglomerative hierarchical clustering is based on the initial assumption that points are individual clusters. During the iterations the closest pair of clusters is being merged together. This requires defining a notion of cluster proximity. This practically means defining minimum distance between clusters to stop merging clusters and get the right number of valid clusters. There is also the Unweighted Pair Group Method using Arithmetic averages (UPGMA), which takes the number of points into consideration while merging, but the computation of this approach is also very intensive.

The initialization process has to be as efficient as possible; the main reason for this is to do not lose necessary information for efficient tracking. If we had a webcam, which is capable of 30fps, we would get a new frame every 33ms. If our initialization method took 300ms we could lose 9 frames and the location of the initial features could change a lot during that time and this could be the reason of tracking failure.

### 3.2 DBSCAN

Density-Based Spatial Clustering of Applications with Noise [7] is a very good unsupervised algorithm. DBSCAN requires the user to specify two parameters - the radius *Eps* of the neighborhood of a point and the minimum number of points *MinPts* in the neighborhood. Iteration starts from single points. While DBSCAN can find clusters with arbitrary shapes (Figure 4), it suffers from a number of problems. DBSCAN is very sensitive to the parameters Eps and MinPts, which in turn, are difficult to determine. DBSCAN also suffers from the robustness problems, in case there is a dense string of points connecting two clusters, DBSCAN could end up merging these two clusters. Also, DBSCAN does not perform any sort of preclustering and executes directly on the entire database and that can be sometimes compute-intensive [2]. The main question about this algorithm is the efficiency problem. It is not designed to be used at computer vision and the possibility of merging hands with head, if there is a small linking part between the objects, is not very ideal.

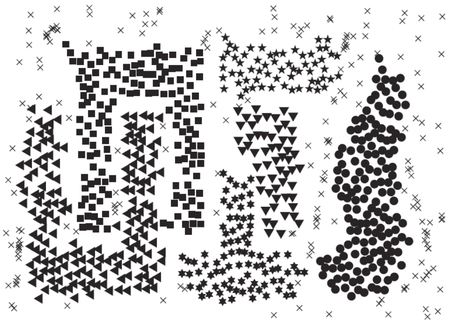


Figure 4: Example of DBSCAN clustering [7].

## 4 Static initialization of multiple flocks

Selecting regions of hand locations manually is not a very practical solution (mostly when there are two hands to track, you do not have the third hand to press the key). Selecting the region for histogram calculation still remains, because the whole KLT features tracking runs on the back-projected image. KLT features are detected all over the segmented image, but at the same time even on the noise. None of the studied algorithms fits our needs, and that is the reason why we created median based static initialization process. The proposed algorithm is able to detect the correct number of flocks (unsupervised learning). Skeleton of the proposed algorithm is presented in Listing 1. The advantage against DBSCAN is that every point is visited only once and it should be more efficient for our kind of datasets. The sample of convergence of the median locations with the final result of the given algorithm is shown in Figure 5.

There are 3 important parameters in this algorithm. The point is considered in range of a given median if it is closer than the maximum distance threshold, which is already defined for FoF algorithm (80 pixels in our case). For merging clusters it can be half of this threshold. Recalculation of the medians is done every $10^{th}$ point, because the possibility that 3 points from 10 will be added to the same cluster is high and the location of median can be changed significantly in such a case. We are planning to optimize this part as future work, finding the more appropriate parameters by experiments or changing the event of recalculation the medians.

Listing 1: Skeleton of the algorithm.

```
Detect KLT features all around the image.
Wait few frames without determining median. //most of the noise features will
    be lost during short period of time (from observation we decided to wait
    30 frames)
Initial the field of medians and field of points for every median.
First point is automatically median.
For every point
        For every median
                If point is in range of given median
                        Add it to that median field
                Else
                        Create new median
        After X points recalculate all medians
Merge flocks which are too close to each other.
Delete small flocks (noise). //we deleted flocks smaller than 10 points
```

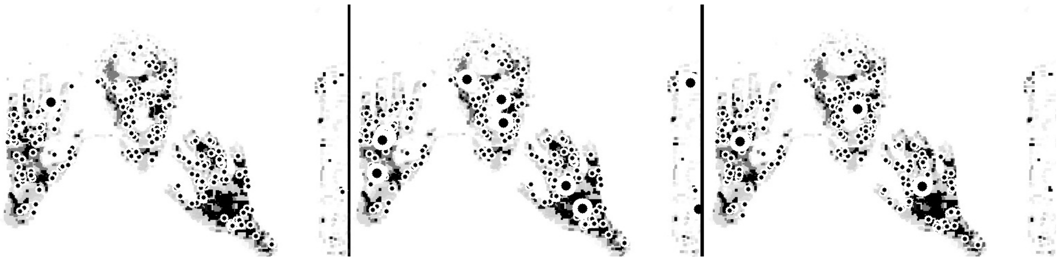


Figure 5: Adding medians, convergence of the medians and the final state of the medians after removing small flocks.

## 5 Dynamic FoF initialization

It is not very comfortable to hold hands up for a longer period of time. We realize that people will put their hands on the table to have rest from time to time. It would be very inconvenient to start initialization every time hands show up in the image. The question is: How do we know when the hands are already there to run the initialization process? The idea of hand detection during tracking is simple. We were inspired by KLT features tracking [6] abilities, where they run KLT features detection only on one side of the video made from a moving car. The results of tracking these features were their trajectories. A new object has to appear in the frame sequence from somewhere. We activate artificial KLT features points around the border of the whole image and after detecting any movement (Figure 6) directed inside the image, these features (squares in Figure 6) are added to the already tracking features (circles). We can detect any movement of skin color object and add KLT features on it. We are still working with back projected image, which in this case is very effective and useful. After getting sufficient number of features, these features are not considered as noise anymore and the new flock with median is created to be tracked.

We tested this method with several videos and we found it vulnerable to the noise located near the border of the frame. If there is a lot of noise, a lot of KLT features can be detected and the new flock is created. An implementation issue of this principle is the frequency of detecting new features. When the KLT features detection is too often, one objects can obtain a lot of features (even hundreds), which is causing slower response of the tracker. We run artificial features tracking every $10^{th}$ frame. During rapid movements the number of features is adequate, but during slower movements a lot of features were detected and the number is being adjusted.

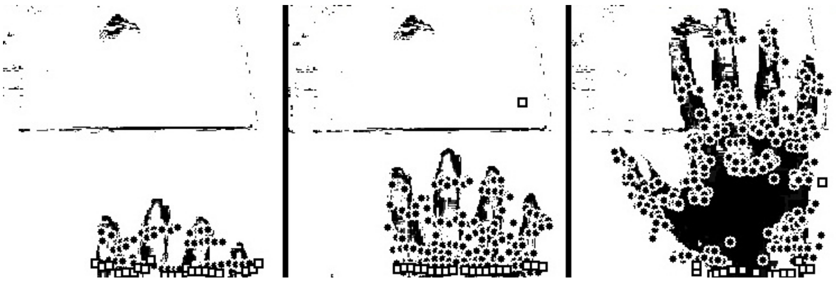


Figure 6: Hand is entering the view angle of the camera and KLT features are adding onto the hand from the lower border of the image.

## 6 Conclusion and future work

Proposed static initialization algorithm is a good idea how to set up FoF tracking for multiple objects, but there is a need to optimize its parameters and define more precisely when the median/s should be recalculated to achieve faster convergence. This algorithm can possibly fail when two objects (hand and the other hand or head) are too close together and they can be merged into one flock. Solution to this task can be done as future work. Second dynamic detection method is very simple and that is why it is very efficient and able to run side by side FoF tracking. The only issue is bigger noise located near the border, which can cause false positive cases.

## References

[1] Fogelton, A.: Real-time hand tracking using Flocks of features. In: *Proceedings of CESCG 2011*, Viničné, 2011.

[2] Guha, S., Rastogi, R., Shim, K.: CURE: an efficient clustering algorithm for large databases. *SIGMOD Rec.*, 1998, vol. 27, pp. 73–84.

[3] Kölsch, M., Turk, M.: Fast 2D Hand Tracking with Flocks of Features and Multi-Cue Integration. In: *Computer Vision and Pattern Recognition Workshop, 2004. CVPRW '04. Conference on*, 2004, pp. 158 – 158.

[4] Reynolds, C.W.: Flocks, herds, and schools: A distributed behavioral model. *Computer Graphics*, 1987, vol. 21, no. 4, pp. 25–34.

[5] Shi, J., Tomasi, C.: Good features to track. In: *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR '94., 1994 IEEE Computer Society Conference on*, 1994, pp. 593 –600.

[6] Sudipta N Sinha, Jan-Michael Frahm, M.P., Genc, Y.: GPU-Based Video Feature Tracking and Matching. In: *EDGE 2006, workshop on Edge Computing Using New Commodity Architectures*, Chapel Hill, 2006.

[7] Tan, P.N., Steinbach, M., Kumar, V.: *Introduction to Data Mining, (First Edition)*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2005.

[8] Tomasi, C., Kanade, T.: Detection and Tracking of Point Features. *Image Rochester NY*, 1991, pp. Technical Report CMU–CS–91–132.