CHASING INSECTS: A SURVEY OF TRACKING ALGORITHMS

PERSIGUIENDO INSECTOS: ALGORITMOS DE TRACKING

S. Frayle-Pérez†, A. Serrano-Muñoz, G. Viera-López and E. Altshuler

Grupo de Sistemas Complejos y Física Estadística, Facultad de Física, Universidad de La Habana, 10400 La Habana, Cuba; sfp932705@gmail.com
† corresponding author

Recibido 15/5/2017; Aceptado 28/6/2017

PACS: Image processing algorithms, 07.05.Pj; Image analysis, 87.57.N-; Movement and locomotion, 87.85.gj; Motion sensors, 07.07.Df

Motion analysis of animals has been a continued challenge for scientists. Traditionally, researchers used non automatic methods to record the animal behaviour [1, 2]. More recently, computer vision development made this task less labor-intensive and results became more accurate. Several tracking algorithms have been employed to capture animal trajectories.

In the case of insects the tracking process is often affected by the interaction between them [3]. The work by Maitra *et al.*, Khan *et al.* and Veeraraghavan *et al.* show different approaches for tracking multiple bees [4–6]. Equivalently Balch *et al.*, Fletcher *et al.* and Ying developed other solutions for tracking multiple ants [7–9]. Other studies limited the amount of insects to one, but tried to capture a very accurate trajectory like Baatrup and Bayley using spiders [10] or Reyes *et al.* using ants [11]. More recently the work of Decamp started breaking the limitations imposed by the camera's field of view by adding a GPS reference to a mobile sensor [12].

We are interested in studying single individuals in non confined areas [13]. Our long term goal is to be able to film the insects with an accurately moving camera. So, if we are able to determine very precisely the position of the insect referred to the camera, and then the position of the camera referred to the land, we will be able to determine the position of the insect referred to the land accurately, even within relatively large distances.

In this work we analyze and compare different computer vision algorithms to detect and track a single insect. Each algorithm is evaluated taking into account the computing time and the possibility of working if the camera position is changed. Algorithms that can successfully carry out the tracking task when the camera position is changed are tested using a mobile camera system. Finally we propose a method for tracking non-flying insects in unconfined regions using a combination of some of the algorithms presented and a mobile camera system.

I. TRACKING ALGORITHMS

We benchmark each algorithm using our own implementations. All the software was written in Python with OpenCV. Independently of the algorithm, for a better performance we do not process every full frame. Instead we analyze a small neighborhood of pixels, centered around the insect, that gets updated from frame to frame rather than the whole image itself. This Region of Interest (ROI) is what our algorithms will be tracking throughout the sequence of image frames. When the insect moves, the estimated position will differ from the last and the ROI will be displaced, in such a way that it is centered in this new position. Studying just a portion of the image considerably speeds up the tracking algorithm.

Once the moving insect is detected, its center of mass must be calculated. This center of mass is given by Equation 1, where m_{pq} denotes the pq^{th} moment and is calculated by means of Equation 2, in which $I(x, y)$ is the image value of the pixel xy. This center of mass is assumed as the estimated position of the insect.

$$C = \left(\frac{m_{10}}{m_{00}}, \frac{m_{01}}{m_{00}}\right) \tag{1}$$

$$m_{pq} = \sum_{x,y} x^p y^q I(x, y) \tag{2}$$

In the following sections we provide a brief description of the tracking algorithms we used.

I.1. Frame Differencing

Motion can be detected by calculating the difference between two consecutive frames [14]. In our proposed solution the frames are converted to gray scale and subtracted from each other, as stated in Equation 3, where $F_i(x, y)$ are the xy pixels of the i^{th} frame and $D_i(x, y)$ the difference. The xy pixels corresponding to regions in the image that stay static have no intensity variations and therefore the difference will be null. On the other hand, those that do experience any sort of movement will result in values different from zero. This algorithm excels in static camera systems but has poor performance in mobile camera systems as the xy pixels in the image vary in intensity while the camera moves.

$$D_i(x, y) = \left|\tfrac{1}{2} \cdot F_i(x, y) - \tfrac{1}{2} \cdot F_{i-1}(x, y)\right| \tag{3}$$

I.2. Background Subtraction

This algorithm starts by modeling a background reference. The presence of moving insects will be determined by subtracting the current frame from this model. Variations in this difference in terms of pixel intensities indicate existence of motion. The proposed algorithm is the one described in [15]. It is ideal for a static camera system, where the background can be perfectly modeled as it is not constantly changing.

I.3. Optical Flow

This algorithm calculates image optical flow field, as described in [16], where each element of the field is a vector containing displacement information of a pixel. The angle of this vector indicates the moving direction of the pixel and the magnitude responds to the distance it moved. The region of pixels in the ROI that have more optical flow are due to the movement of the insect. A major drawback is that any object inside the ROI bigger in dimensions than the insect will have more optical flow once the camera moves, even though if it remains static.

I.4. Corner Detection

Corners and edges in an image are boundaries that imply intensity variations. This algorithm tracks insects by localizing parts in the ROI where the highest variation in intensity is found. The corner and edge detection is described in [17]. It is not suitable for video frames that have large pixel intensity changes because other corners and edges may be found inside the ROI that do not correspond to the tracked insect.

I.5. Color Matching

The low computational cost of color based tracking algorithms makes color an interesting feature to exploit [14]. In our proposed algorithm, the range of the pixel intensities of the tracked insect must be provided. In each frame the ROI is segmented in two regions: a foreground region that contains the intensities of the insect, and a background one that has pixels with values out of that range. It is appropriate only when the insect and the background are highly color contrasted and the illumination conditions of the scene are kept relatively constant.

I.6. Simple Template Matching

Template Matching is a technique to find a template image within a larger one. The work flow is to fit the template image in all possible positions in the source image and find the location where it best matches pixel by pixel [18]. Our proposed algorithm slides the template, which contains the insect, over the current frame and finds the pixel location in the ROI whose neighborhood maximizes the template match. This algorithm has a high computational cost and is affected by insect rotations.

I.7. ORB Descriptor Matching

ORB (Oriented FAST and Rotated BRIEF) is a fast binary descriptor, that is rotation invariant and resistant to noise, used in feature matching [19]. In our work a template image containing the insect must be provided to be searched in the ROI. The algorithm takes the descriptor of one feature in the template and matches it with all other features in the ROI; the best match is kept. This is done for all features in the template. On some frames one or more features may be found, on others, none. However the latter holds on such rare occasions. To estimate the position of the insect we average the coordinates that the matched features occupy. This is an expensive algorithm but it is very robust.

I.8. Correlation Tracker

Feature detection and matching between two signals has been approached using correlation filters [20]. These algorithms have a good performance and are robust enough to deal with partial occlusion as well as variations in rotation, scale and lighting conditions. The algorithm employed is the one described in [21].

II. PERFORMANCE WITH A FIXED CAMERA SYSTEM

In order to visualize the differences between the results of every single algorithm we analyze the trajectories obtained for the same video. The following results were achieved tracking an ant from the species *Atta insularis* [22, 23] in a video that was recorded with a camera fixed at 1.80 m of height. Keeping the camera static limits the trajectory of the ant to a particular region. Once the insect escapes the field of view of the camera, the experiment is over. Figure 1 shows the trajectories estimated by the algorithms previously described for this video. All the paths were plotted using raw data, there is no filtering or processing applied at all. The maximum uncertainty between the estimated paths is under 0.015 m. Filtering techniques can be applied to reduce that uncertainty. Keeping the camera fixed limits the tracking area depending on the height of the camera. Increasing that area (rising the camera) will decrease the tracking resolution.

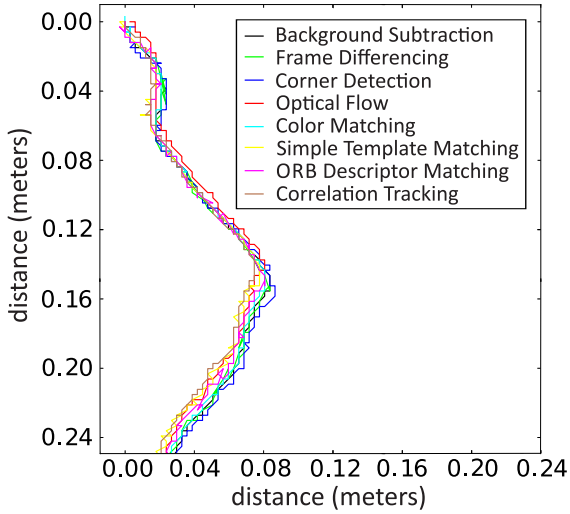

Figure 1. Trajectories of a moving ant (*Atta insulariis*) estimated by the tracking algorithms for a fixed camera video.

III. PERFORMANCE WITH A MOBILE CAMERA

In addition to a fixed camera system, where only a fixed region can be studied, mobile camera systems allow to increase the tracking arena without losing tracking resolution. We built a one degree of freedom mobile camera system placed at a height of 0.30 m for testing the algorithms. We also plan to build a more sophisticated one that allows to determine with high precision the camera position in two dimensions. The camera can move in the same general direction of the insect motion as it is perceived to abandon the original field of view.

The accuracy in the estimation of the camera movements is a key factor for long distance tracking, due to the incremental errors that may be induced. We store the position of the camera on each frame to be able to transform the position of the insect referred to the camera into ground coordinates. Figure 2 shows the trajectories resulting from applying the tracking algorithms to the video of an moving ant, and once it is near the end of the field of view of the camera, the camera was moved 0.10 m to keep it inside.

Notice that when the insect turns the uncertainty between the estimated position given by the algorithms increases. This is caused by the different nature of the algorithms and not by the movement of the camera. In fact, increasing the tracking resolution will make the algorithms estimate different sized shapes of the same moving insect and therefore the calculated center of mass will differ. Anyway, the maximum uncertainty is below 0.015 m, which is of the order of the insect's size, and similar to the one obtained with the fixed camera setup.

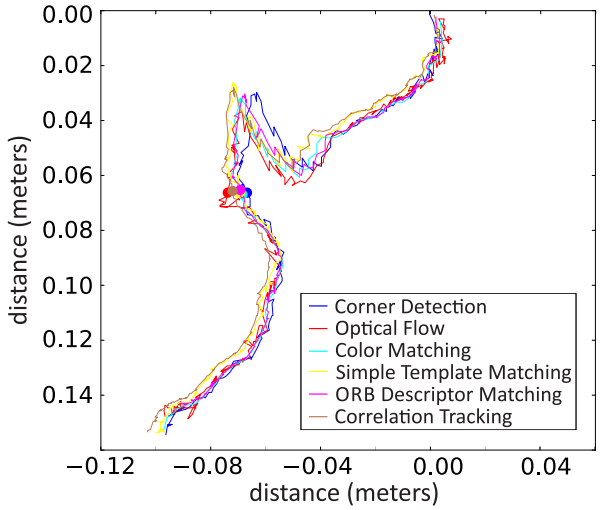

Figure 2. Trajectories estimated by the tracking algorithms for a mobile camera video. The dots in the image indicate the frame in which the camera changed its position.

The processing rate of the algorithms in frames per second calculated averaging over 15 videos is given in Table 1.

Table 1. Average processing rate of the algorithms

Algorithm	**Processing rate (fps)**
Frame Differencing	207
Background Subtraction	53
Color Matching	224
Corner Detection	223
Optical Flow	202
Simple Template Matching	177
ORB Descriptor Matching	171
Correlation Tracker	56

IV. SUMMARY

We have analyzed and compared different tracking algorithms that could be used for studying the trajectory of a single insect. The ones that can handle camera movements were tested using a mobile camera system. Background Subtraction and Frame Differencing failed this purpose. Results of the trajectories estimated were presented. Since Frame Differencing does not need to model or learn a background image, we propose to use it while the camera remains static. Once it needs to be moved given that the insect is reaching the limits of the field of view of the camera, ORB Descriptor Matching should be employed because of its robustness and high processing rate.

REFERENCES

[1] L. P. Noldus, Behav. Res. Methods. 23, 415 (1991).

[2] L. P. Noldus, R. J. Trienes, A. H. Hendriksen, H. Jansen and R. G. Jansen, Behav. Res. Methods. 32, 197 (2000).

[3] A. Reyes, F. Tejera and E. Altshuler, Rev. Cubana Fis. 33, 44 (2016).

[4] P. Maitra, S. Schneider and M. C. Shin, Applications of Computer Vision (WACV), 2009 Workshop on, 1 (IEEE, 2009).

[5] Z. Khan, T. Balch and F. Dellaert, IEEE Trans. Pattern Anal. Mach. Intell. 27, 1805 (2005).

[6] A. Veeraraghavan, R. Chellappa and M. Srinivasan, IEEE Trans. Pattern Anal. Mach. Intell. 30, 463 (2008).

[7] T. Balch, Z. Khan and M. Veloso, Proceedings of the fifth international conference on Autonomous agents, 521 (ACM, 2001).

[8] M. Fletcher, A. Dornhaus and M. C. Shin, Applications of Computer Vision (WACV), 2011 IEEE Workshop on, 570 (IEEE, 2011).

[9] F. Ying, Visual ants tracking, Ph.D. thesis, University of Bristol (2004).

[10] E. Baatrup and M. Bayley, Physiol. Behav. 54, 83 (1993).

[11] A. Reyes González, Hacia una comprensión cuantitativa de la "exploración libre" en insectos sociales., B.S. Thesis, Universidad de la Habana (2016).

[12] L. Decamp, Tracking ants from video data: accurately retrieving the pose of the hand-held camera., M.Sc. Thesis, University of Edinburgh, UK (2015).

[13] A. Reyes, G. Rodriguez and E. Altshuler, Rev. Cubana Fis. 33, 134 (2016).

[14] H. S. Parekh, D. G. Thakore and U. K. Jaliya, IJIRCCE 2 (2014).
[15] P. KaewTraKulPong and R. Bowden, Video-based surveillance systems, 135 (Springer, 2002).
[16] G. Farneback (2003). URL `http://www.isy.liu.se/cvl/`.
[17] C. Harris and M. Stephens, Alvey vision conference, volume 15, 10 (Citeseer, 1988).
[18] J. J. Athanesious and P. Suresh, IJARCET 2, 242 (2013).
[19] E. Rublee, V. Rabaud, K. Konolige and G. Bradski, Computer Vision (ICCV), 2011 IEEE International Conference on, 2564 (IEEE, 2011).
[20] H. Kiani Galoogahi, T. Sim and S. Lucey, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 4630 (2015).
[21] M. Danelljan, G. Häger, F. Khan and M. Felsberg, British Machine Vision Conference, Nottingham, September 1, 2014 (BMVA Press, 2014).
[22] C. Noda, J. Fernández, C. Pérez-Penichet and E. Altshuler, Rev. Sci. Inst. 77, 126102 (2006).
[23] S. Nicolis, J. Fernández, C. Pérez-Penichet, C. Noda, F. Tejera, O. Ramos, D. J. T. Sumpter and E. Altshuler, Phys. Rev. Lett. 110, 268104 (2013).