ESTIMATING ORIENTATION IN TRACKING INDIVIDUALS OF FLYING SWARMS

*Xi En Cheng, Shuo Hong Wang, Yan Qiu Chen**

School of Computer Science, Shanghai Key Laboratory of Intelligent Information Processing
Fudan University, Shanghai, China

ABSTRACT

Multi-camera three-dimensional (3D) tracking technology is the most effective gateway to acquire quantitative motion data for studying behaviors of flying swarms. While the state-of-the-art tracking methods usually estimate positions of flying targets, here we present the 3D tracking method which estimates both a target's position and its orientation for tracking individuals of flying swarms, in which videos captured by at least three synchronized and calibrated cameras are necessary. Experiments show the proposed method outperforms the state-of-the-art methods.

***Index Terms*—** Target swarm, 3D tracking, orientation, Rao-Blackwellization, Kalman filtering

1. INTRODUCTION

Motion measurement for animals in swarms has been enabled by recent advances in video tracking [1–3]. The existing 3D tracking methods have demonstrated success in tracking flying targets [4–7]. As tracking individuals of flying swarms using videos captured by multiple cameras, the 3D motion state estimation is a challenging problem: the entire swarm has to be filmed in all cameras' field-of-view, and thus makes the observation of each target very coarse. Each target only takes up a small image area and resembles each others. That is, there is little visual cues for distinguishing and associating observations with targets. Though the state-of-the-art methods report results on tracking tens to hundreds of flying targets [4–10], they usually estimate positions of a flying target through time. There are few works focusing on investigating the ability of estimating a target's flight attitude in a 3D tracking system.

As a flying swarm (that usually contains hundreds of individuals) is filmed by multiple ($\geq$ 3) synchronized and calibrated cameras, we aim to track individuals of the flying swarm and output sequential motion states including position and orientation. The main contribution is the 3D position and orientation tracking algorithm, in which the Rao-Blackwellization technology is adopted to partition the state space of a target's position and of a target's orientation. Meanwhile, we developed an algorithm for updating the 3D motion state using two-dimensional (2D) observations across views.

*Corresponding author. Thanks to National Natural Science Foundation of China, Grant No. 61175036 for funding.

2. MODELING A FLYING TARGET

A flying target (*e.g.* a fly, as shown in Fig. 1a) can be briefly defined by an articulated model, as shown in Fig. 1b, if we assume its body undergoes rigid body motion. Here a target's body includes its head, throat, and abdomen. The tracking system usually focuses on the motion state of a target's body motion. That is, the wings can be neglected from the articulated model. At this point, we can use an ellipsoid to model a flying target. The flight attitude of a flying target is characterized by *pitch*, *yaw*, and *roll* angles against the body-frame, as shown in Fig. 1c.

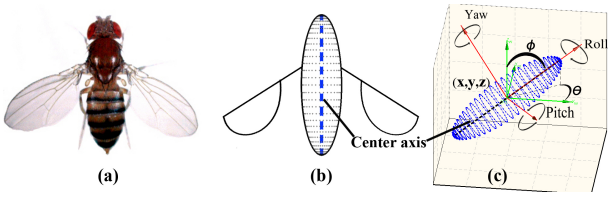

Fig. 1. Modeling a flying target.

Most animals capable of flight have bilateral symmetric (or plane symmetric) body. Here we assume that a target's body is radial symmetric (or axial symmetric). Following this assumption, an axis of symmetry (*center axis*) is found in the body, *e.g.* the *mean axis* of the ellipsoid. At this point, given the *center axis*, a target's orientation can be defined by Euler angles of its *center axis*, *polar* angle ϕ and *azimuthal* angle θ against the reference coordinate frame in Euclid, as shown in Fig. 1c. Therefore, instead of estimating a target's flight attitude, we estimate a target's orientation in this work. A flying target's motion state is thereby defined by its position (x, y, z) and orientation (θ, ϕ). That is, a flying target is modeled as a line-segment (the center axis), parameterized by

$$\{x, y, z, \theta, \phi, \tilde{l}\} \tag{1}$$

where $\tilde{l}$ is a constant scalar value defining the length of body.

3. THE 3D-POT ALGORITHM

The algorithm we propose is the 3D position and orientation tracking (3D-POT) algorithm. The 3D-POT algorithm treats each target as a dynamic system and thus adopts the Bayesian inference framework to solve the problem.

3.1. State and state space partition

By definition, a flying target is parameterized in (1), the system state can thereby be defined as

$$\{x, y, z, x^-, y^-, z^-, \theta, \phi\} \tag{2}$$

where the "x^-" terms are the position of the target at previous moment. In this study, the state transition does not rely on the velocity prior. It is important for tracking targets in a swarm since the targets may take complex maneuvers and thus violate the smooth motion assumption.

It is known that the polar angle (ϕ) of the orientation is in a damped range about 45° to the negative direction of gravity for the reason of aerodynamics [11]. It means that ϕ changes independently of the position movement. But, for the reason of kinematic, the azimuthal angle (θ) of the orientation has to align to the azimuthal angle of the target's motion direction in most of time. That is, θ changes conditionally on the position movement. The state can thereby be partitioned into two groups: the position state X and the orientation state O, which are denoted by

$$X = \{x, y, z, x^-, y^-, z^-\}, O = \{\phi, \theta\} \tag{3}$$

With the above idea, this facilitates the different dynamic models of these two groups. The relationship between these two groups is that changes of orientation states are conditioned on the changes of position states.

3.2. State transition and observation

The evolution of the position state from moment $t-1$ to moment t is controlled by a non-linear motion model, which is defined by

$$X_t = \mathbf{f}(X_{t-1}) + \upsilon_{t-1} \tag{4}$$

where $\mathbf{f}(X_{t-1})$ denotes the dynamic function and υ_{t-1} denotes the state transition noise. Conditioned on the position state at moment t, the evolution of the orientation state from moment $t-1$ to moment t is controlled by a linear and Gaussian motion model:

$$O_t = \mathbf{G}O_{t-1} + \nu_{t-1}, \nu_{t-1} \sim \mathcal{N}(0, \mathbf{Q}) \tag{5}$$

where $\mathbf{G}$ denotes the linear transition function and ν_{t-1} denotes the spherical wrapped Gaussian noise.

In the video tracking system, images (frames of a video) from camera views are the raw input of the tracking system. An image includes the observation of many targets

Algorithm 1 The 3D-POT Algorithm.

Input: The target's information at time $t-1$, *e.g.* state, appearance; particles at time $t-1$:$\{s^i_{t-1}, i = 1..N\}$; measurements at moment t: $\{\chi_{v,j}, v = 1..v_n, j = 1..j_{v,n}\}$.

Output: Particles at moment t, and the expectation, $\hat{s}_t$.

1: **for** $i = 1$ to N **do**
2: Initial particle $s^i_t = (*, *, *)$;
3: Sample the position state of $s^i_t = (X^i_t, *, *)$;
4: Kalman prediction, $s^i_t(X^i_t, \mu^i_{t|t-1}, \sigma^i_{t|t-1})$ (7);
5: Associate measurements with s^i_t (8);
6: Evaluate "alive" particle's weight, w^i_t;
7: **end for**
8: Normalize $w^i_t = \dfrac{w^i_t}{\sum_{i=1}^N w^i_t}, i = 1..N$;
9: Particles resampling;
10: **for** $i = 1$ to N **do**
11: Kalman update, $s^i_t(X^i_t, \mu^i_t, \sigma^i_t)$ (9,10);
12: **end for**
13: **return** $\{s^i_t, i = 1..N\}$,$s^i_t = (X^i_t, \mu^i_t, \sigma^i_t)$ and $\hat{s}_t = E(s_t) = \sum_{i=1}^N s^i_t * w^i_t$;

"observed" by the camera. The observation of a certain target at moment t, Z_t, is usually the high level data (***measurement***) extracted from the raw input. Since we represent a target with an ellipsoid, a measurement $\chi = \{b, e(b)\}$ of the target includes two components: (i) a blob b which denotes the image patch of the projection of the ellipsoid; (ii) an ellipse $e(b)$ which denotes the silhouette of the ellipsoid and is an ellipse fitted to the blob b. The two components of a measurement reflect the two aspects of an observation: appearance and orientation. Measurements at each moment are defined as $\{\chi_{v,j}, v \in \{1, ..., v_n\},\ j \in \{1, ..., j_{v,n}\}\}$, where v denotes a camera view.

3.3. Algorithm overview

The aim of recursively estimating the posterior density of position states can be accomplished using particle filter. At the same time, conditioned on the nonlinear position states, Kalman filter can be used for obtaining orientation states. That is, the posterior of the system is factorised by

$$p(X_t, O_t|Z_t) = \underbrace{p(X_t|Z_t)}_{PF}\underbrace{p(O_t|X_t, Z_t)}_{KF} \tag{6}$$

That is, the Rao-Blackwellization [12] is employed to solve the system's posterior distribution.

Briefly, the 3D-POT algorithm approximates the system's posterior density at moment t by N weighted particles: $\{s^i_t, w^i_t|i = 1..N\}$. Each particle contains both the position state and the parametric representation of the distribution of the orientation states, which consists of the mean value of the orientation states $\mu^i_t = (\phi^i_t, \theta^i_t)^{\mathrm{T}}$ and the error covariance σ^i_t.

That is, each particle is defined by a triplet $s_t^i = (X_t^i, \mu_t^i, \sigma_t^i)$. Algorithm 1 depicts the 3D-POT algorithm, in which the position tracking component is adapted from previous work [6].

3.4. Kalman prediction

The evolution of the orientation state is controlled by the linear model defined by (5). By using straight forward application of the Kalman filter, the prediction of mean $\mu_{t|t-1}^i$ and error covariance $\sigma_{t|t-1}^i$ are computed as

$$\begin{array}{rcl} \mu_{t|t-1}^i & = & \mathbf{G}\mu_{t-1}^i + \nu_{t-1}, \nu_{t-1} \sim \mathcal{N}(0, \mathbf{Q}) \\ \sigma_{t|t-1}^i & = & \mathbf{G}\sigma_{t-1}^i \mathbf{G}^{\mathrm{T}} + \mathbf{Q} \end{array} \tag{7}$$

where ν_{t-1} denotes the spherical wrapped 2D Gaussian noise. At this point, we have the particle $s_t^i = (X_t^i, \mu_{t|t-1}^i, \sigma_{t|t-1}^i)$, in which $\mu_{t|t-1}^i = (\theta_{t|t-1}^i, \phi_{t|t-1}^i)^{\mathbf{T}}$.

3.5. Measurement Association

At each moment, measurements are extracted from images. It is necessary to determine the association between a certain target and a measurement in a certain camera view. That is, a measurement has to be assigned to a target if the target is "observed" in that camera view. This is the task of measurement association, and we propose the pixels occupancy test algorithm for solving the task. The algorithm is derived from the idea of probability gating [8,13], and simultaneously solves the association problem across all camera views. By definition, the algorithm suggests that any measurements in a certain image region are associated with a certain target with identical probability. In order to determine the probability gate for a target, we adopt a sphere to represent a target in 3D space and project the sphere onto camera views, and thus the sphere's image serves as the probability gate. Let $\Gamma(s_t^i)$ define the association between the particle $s_t^i = (X_t^i, \mu_{t|t-1}^i, \sigma_{t|t-1}^i)$ and measurements $\chi_{v,j}, v = 1..v_n, j = 1..j_{v,n}$,

$$\begin{array}{l} \Gamma(s_t^i) = \{(\chi_{v,j}, \eta) \mid \eta > \tilde{\eta}, v = 1..v_n, j = 1..j_{v,n}\} \\ \quad \eta = \dfrac{\kappa(\Omega(X_t^i, \chi_{v,j}))}{\kappa(b_{v,j})} \\ \Omega(X_t^i, \chi_{v,j}) = \{\mathbf{P}^v \Upsilon^3\} \bigcap b_{v,j} \end{array} \tag{8}$$

where $\kappa(\cdot)$ defines the function counting the number of pixels, and $b_{v,j}$ denotes the blob component of the measurement $\chi_{v,j}$. Here Υ^3 denotes the discrete points sampled on the sphere which locates at $X_t^i(x_t^i, y_t^i, z_t^i)$ with the diameter equal to $\tilde{l}$, and $\mathbf{P}^v$ is the projection matrix of camera v; and thereby, $\Omega(X_t^i, \chi_{v,j})$ defines the common pixels. In (8), $\tilde{\eta}$ is a scalar positive-valued percolation threshold which varies from 0 to 1 for leveraging the restriction on association. Larger $\tilde{\eta}$ means more restrictive association. Particles which successfully associate with measurements in all views are "alive" particles; others are "dead" particles and thereby be neglected.

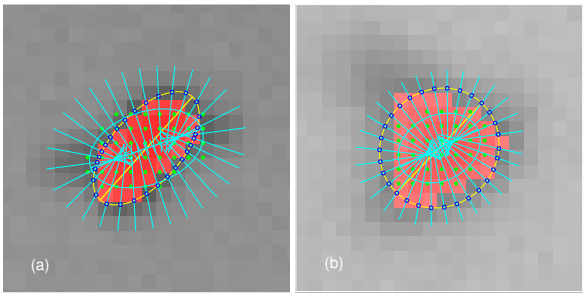

Fig. 2. The blob b_v is marked in red and the ellipse $e(b_v)$ is yellow. The cyan ellipse is $e(\mathbf{P}^v \Upsilon^3)$ and the cyan lines are normal lines at N_e points on the ellipse. The blue circles denote cross points on $e(b_v)$. Here only two views are shown.

3.6. Kalman updating

By using straight forward application of the Kalman filter, updating the mean μ_t^i and error covariance σ_t^i for the orientation state are computed as

$$\begin{array}{rcl} \mu_t^i = & \mu_{t|t-1}^i + \sigma_{t|t-1}^i \mathbf{K}_t^i \Delta O_t \\ \sigma_t^i = & \sigma_{t|t-1}^i - \mathbf{K}_t^i \mathbf{C} \sigma_{t|t-1}^i \\ \mathbf{K}_t^i = & \sigma_{t|t-1}^i \mathbf{C}^{\mathrm{T}} (\mathbf{C}\sigma_{t|t-1}^i \mathbf{C}^{\mathrm{T}} + \mathbf{R})^{-1}, \end{array} \tag{9}$$

where $\mathbf{R}$ is the covariance of the Gaussian noise for observing the orientation state and $\mathbf{C}$ denotes the observation matrix. It shows that updating the mean value μ_t^i depends on computing ΔO_t, which is the difference between the predicted orientation and the observed orientation at moment t. However, since we only have 2D observations a target's orientation is unable to be observed. That is, it is infeasible to compute ΔO_t directly. In this paper, we propose an algorithm for computing ΔO_t using 2D measurements across views.

According to the aforementioned, the target represented by particle s_t^i has the motion state $(x_t^i, y_t^i, z_t^i, \theta_{t|t-1}^i, \phi_{t|t-1}^i)$. Therefore, we can generate an ellipsoid to represent the target where its center locates at (x_t^i, y_t^i, z_t^i) and the direction of its center axis is defined by the orientation $(\theta_{t|t-1}^i, \phi_{t|t-1}^i)$, and the length of its center axis equals to $\tilde{l}$. Let Υ^3 denote the points sampled on the ellipsoid. The difference between the predicted orientation and the observed orientation, ΔO_t, is thereby computed as

$$\begin{array}{l} \Delta O_t \propto exp(-\sum_{v=1}^{v_n} d(e(b_v), e(\mathbf{P}^v \Upsilon^3)) - \hat{d}), \\ \qquad \chi_v(b_v, e(b_v)) \in \Gamma(s_t^i) \end{array} \tag{10}$$

where the function $d(\cdot)$ computes the distance between the ellipse $e(b_v)$ and the ellipse, $e(\mathbf{P}^v \Upsilon^3)$, fitted to the points projected from Υ^3. Here $\hat{d}$ denotes the standardized distance between two ellipses. The function $d(\cdot)$ computes the accumulative pairwise distance among cross points on lines and two ellipses. Fig. 2 depicts the procedure. These lines are norm lines at N_e points which are uniformly sampled on the

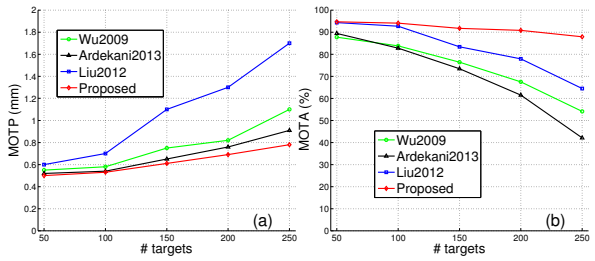

Fig. 3. Comparison of the proposed method with state-of-the-art methods on position estimation. Lower is better for MOTP scores and higher is better for MOTA scores [15].

ellipse $e(\mathbf{P}^v\Upsilon^3)$. We choose $N_e = 20$ and have done a parametric study to find the most appropriate parameter range for $\hat{d}$, and we found that $\hat{d} = 50$ works better for the experiments.

4. EXPERIMENTS

The real-world swarms usually have hundreds of individuals with similar appearance, manually creating the 3D ground truth is infeasible: (i) each target only has small image area and resembles each other; (ii) there are only 2D observations. Using simulation data to evaluate performance is more preferable for methods on tracking large number of targets [5–7, 14]. By carefully designed simulation data we evaluate performance of the proposed method and compare with the state-of-the-art methods: Wu2009 [5], Liu2012 [6], Ardekani2013 [7]. We adopt the CLEAR MOT metrics [15] to measure the performance of position estimation and choose the body length $\tilde{l} = 3$ mm as the matching criterion.

4.1. Dataset

The sequential 3D positions of each target are generated using a model adapted from boids [16]. Then the motion direction $M(\theta, \phi)$ is computed at each moment and the orientation $O(\theta, \phi)$ is thereby computed as

$$O = \begin{pmatrix} M(\theta) \\ \frac{\pi}{4} \end{pmatrix} + n, n \sim \mathcal{N}(0, 1) \qquad (11)$$

where n denotes the white noise. Given a target's position and orientation, a shape is generated. At each moment, all targets represented by shapes are filmed by simulated cameras. We applied the "Machine Vision Toolbox" [17] to simulate the three-camera system, and generated five simulation datasets (*D1-D5*) with the number of targets $N = \{50, 100, 150, 200, 250\}$, respectively.

4.2. Performance evaluation

Compared to the proposed method, Fig. 3a shows that Wu2009 and Ardekani2013 obtain nearly same scores (low MOTP score) on *D1* and *D2* but worse results on *D3-D5*. Wu2009 tracks targets using 2D observations and then reconstructs 3D trajectories using 2D trajectories across views. Its precision mostly depends on the target detection at the first step, and thereby decreases as the population increases. Ardekani2013 reconstructs 3D positions as candidates and then tracks them in 3D. While several targets overlapped in images, the candidates are thereby drifted.

Fig. 3b shows Liu2012 obtains almost same scores (high MOTA score) on *D1* and *D2*, as the proposed method does; but Liu2012 shows fastly decreasing as the population increases. Liu2012 has no data association approach and accepts particles even if they are only "observed" in one camera view, and thus obviously increases the possibility of trackers being distracted by other targets and thus decreases its robustness.

While evaluate the performance reported in Table 1, the matching criterion is that the angle between an estimated orientation and the ground truth is less than 5°. Though the performance is dominated by the position estimation, the Kalman filter's ability of smoothness poses negative effects. It usually delays the tracked orientation while a target takes sharp turn, as shown in Fig. 4 on real-world data.

Table 1. Performance on orientation estimation

Dataset	*D1*	*D2*	*D3*	*D4*	*D5*
Precision	0.93	0.83	0.84	0.80	0.73
Recall	0.84	0.76	0.73	0.69	0.64
F1-Measure	0.88	0.79	0.78	0.74	0.68

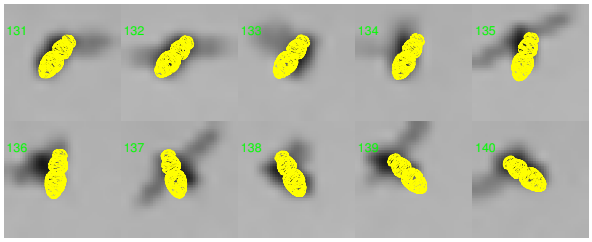

Fig. 4. The sharp turn in 10 frames (0.1 sec).

5. CONCLUSION

The proposed tracking method performs well even though the raw observation is very limited image areas for flying targets. The association algorithm guarantees the accuracy and robustness of the proposed tracking method, and the Kalman updating algorithm enables tracking a target's orientation. Moreover, we have successfully tracked ≈ 700 flies flying in a flight arena (data not presented).

6. REFERENCES

[1] K. Branson, A. A. Robie, J. Bender, P. Perona, and M. H. Dickinson, "High-throughput ethomics in large groups of *Drosophila*," *Nat Meth*, vol. 6, no. 6, pp. 451–457, 2009.

[2] A. I. Dell, J. A. Bender, K. Branson, I. D. Couzin, G. G. de Polavieja, L. P.J.J. Noldus, et al., "Automated image-based tracking and its application in ecology," *Trends in Ecology & Evolution*, vol. 29, no. 7, pp. 417 – 428, 2014.

[3] S. B. Rosenthal, C. R. Twomey, A. T. Hartnett, H. S. Wu, and I. D. Couzin, "Revealing the hidden networks of interaction in mobile animal groups allows prediction of complex behavioral contagion," *Proceedings of the National Academy of Sciences*, vol. 112, no. 15, pp. 4690–4695, 2015.

[4] D. Zou, Q. Zhao, H. S. Wu, and Y. Q. Chen, "Reconstructing 3d motion trajectories of particle swarms by global correspondence selection," in *Computer Vision (ICCV), 2009 IEEE International Conference on*. IEEE, 2009, pp. 1578–1585.

[5] Z. Wu, N. I. Hristov, T. L. Hedrick, T. H. Kunz, and M. Betke, "Tracking a large number of objects from multiple views," in *Computer Vision (ICCV), 2009 IEEE International Conference on*. IEEE, 2009, pp. 1546–1553.

[6] Y. Liu, H. Li, and Y. Chen, *Automatic Tracking of a Large Number of Moving Targets in 3D*, vol. 7575 of *Lecture Notes in Computer Science*, chapter 52, pp. 730–742, Springer Berlin Heidelberg, 2012.

[7] R. Ardekani, A. Biyani, J. E. Dalton, J. B. Saltz, M. N. Arbeitman, J. Tower, et al., "Three-dimensional tracking and behaviour monitoring of multiple fruit flies," *J. R. Soc. Interface*, vol. 10, no. 78, pp. 1–13, 2013.

[8] M. Betke, D. E. Hirsh, A. Bagchi, N. I. Hristov, N. C. Makris, and T. H. Kunz, "Tracking large variable numbers of objects in clutter," in *Computer Vision and Pattern Recognition (CVPR), 2007 IEEE Conference on*. IEEE, 2007, pp. 1–8.

[9] H. S. Wu, Q. Zhao, D. Zou, and Y.Q. Chen, "Automated 3d trajectory measuring of large numbers of moving particles," *Optics Express*, vol. 19, no. 8, pp. 7646–7663, 2011.

[10] X. E. Cheng, Z. M. Qian, S. H. Wang, N. Jiang, A. Guo, and Y. Q. Chen, "A novel method for tracking individuals of fruit fly swarms flying in a laboratory flight arena," *PLoS ONE*, vol. 10, no. 6, pp. e0129657, 2015.

[11] S. N. Fry, R. Sayaman, and M. H. Dickinson, "The aerodynamics of free-flight maneuvers in *Drosophila*," *Science*, vol. 300, no. 5618, pp. 495–498, 2003.

[12] T. Schon, F. Gustafsson, and P. J. Nordlund, "Marginalized particle filters for mixed linear/nonlinear state-space models," *IEEE Trans. Signal Process.*, vol. 53, no. 7, pp. 2279–2289, 2005.

[13] S. Blackman and R. Popoli, *Design and Analysis of Modern Tracking Systems*, Artech House, 1999.

[14] B. Risse, D. Berh, J. Tao, X. Jiang, R. Klette, and C. Klambt, "Comparison of two 3d tracking paradigms for freely flying insects," *EURASIP J. Image Video Processing*, vol. 2013, no. 1, pp. 1–13, 2013.

[15] K. Bernardin and R. Stiefelhagen, "Evaluating multiple object tracking performance: The clear mot metrics," *EURASIP J. Image Video Processing*, vol. 2008, no. 1, pp. 1–10, 2008.

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

[17] Peter I. Corke, *Robotics, Vision & Control: Fundamental Algorithms in Matlab*, Springer, Berlin, 2011.