Vision-based Target Tracking for Micro Air Vehicles Using Random Set Theory

Neeta Trivedi[a,1], S. Lekshmi[b]
[a]Aeronautical Development Establishment, DRDO, Bangalore
[b]Naval Physics and Oceanographic Laboratory, DRDO, Kochi

Abstract

Micro Air Vehicles (MAV) can deliver their full potential only if they are equipped with autonomous navigation capabilities. Optic flow has emerged as a significant tool for many autonomous navigation tasks. Optic flow fields remain to some extent uncertain because of image noise, lighting changes, low contrast regions, multiple motions in single localized regions and so on. Tracking multiple targets by itself is a challenging problem; it is made more challenging by three factors in the MAV scenario: the uncertainties in optic flow, 6 degrees of freedom (DoF) of movement of the platform and possible 6 DoF of movement of the targets. Finite set statistics provides a unified probabilistic foundation for multisource-multitarget tracking; its computational complexity has been addressed by Probability Hypothesis Density filters and many approximation techniques. We propose a statistical model for tracking multiple objects (targets) from optic flow information obtained from nose-mounted camera. Significant contributions of this paper are enhanced state-transition and measurement models and use of random set theory to track varying number of targets. A lightweight mechanism for data association is also proposed.

Key words: Multitarget tracking, optic flow, random set theory, finite set statistics, computer vision, micro air vehicle

1. INTRODUCTION

Micro Air Vehicles (MAV) that hold the promise of tracking anyone, anywhere, any time have drawn considerable attention in the present scenario of asymmetric and proxy wars. This demand pull is well supported by technology push made possible by the miniaturization of electronics, advances in MEMS technology and the like. However, true potentials of MAV can be exploited only if they can navigate and carry out their mission autonomously.

Vision is a very powerful sensor that can be exploited in many ways. Most present vision navigation solutions have been developed for ground vehicles. Given that MAV (and therefore the sensor) has 6 degrees of freedom (DoF) of movement, any target in the field of view will have 6 DoF of movement relative to the MAV. This makes navigation through target tracking or obstacle avoidance much more challenging. Stereo vision can provide depth perception; however, the weight and size constraints of MAV coupled with the requirement for precision mounting of two sensors for stereo vision have led to optic flow computation on sequence of images from a single onboard camera emerging as a significant alternative for achieving many tasks such as autonomous motion detection, computation of time to collision and object segmentation among others. Methods for robust and fast computation of optic flow have been proposed e.g. in [1,2], but optic flow fields are always subject to some degree of uncertainty because of image noise, lighting changes, low contrast regions, the aperture problem and multiple motions in a single localized region [3] and hence target tracking using optic flow is best modelled in the probabilistic domain.

Email addresses: neeta@ade.drdo.in (First Author),
lekshmis@rediffmail.com (Second Author),
[1]Corresponding author

Formal Bayes modelling has become the accepted standard in single-sensor, single-target research and development [4]. However, tracking multiple targets remains a challenge. Traditional method for tracking multiple targets is to model it as a dynamic system whose order is fixed when there are a fixed number of targets. Autonomous navigation with obstacle avoidance is a typical problem of tracking multiple but unknown numbers of targets where objects may appear, disappear, merge and split in the field of view. Most traditional multitarget tracking formulations involve explicit association between measurements and targets [5]. For a resource-scarce platform like MAV, data association problem is computationally expensive even with relatively small number of targets. Finite set statistics (FISST) [4] is a unified, scientifically defensible probabilistic foundation for multisensor-multitarget tracking. The problem of computational complexity of FISST has been addressed by Probability Hypothesis Density (PHD) filter as an approximation of multi-target filter. The PHD filter has the important property that the integral of PHD over a region in the state space is the expected number of targets within this region [6]. Many improved approximation techniques for FISST are being proposed such as in [7,8].

In this paper we propose a statistical model for tracking multiple objects (targets) from optic flow information obtained from a camera mounted on MAV. Without loss of generality, we assume a nose-mounted camera and related applications. Most significant of the contributions of this paper are models for state-transition and measurement using optic flow information and visual characteristics and use of random set theory/finite set statistics to track varying number of targets for various purposes. We propose components of target state vector ideally suited to address multiple problems such as obstacle avoidance, target tracking etc., and discuss mechanism for extracting these components from the optic flow field and storing them in a way that can be used to carry out multiple tasks with minimum additional overheads. The algorithm assigns unique IDs to targets and maintains history for required period of time; this helps in continuous tracking when there are missed detections/temporary occlusions. We do not propose method to compute optic flow; we assume that a suitable method has been used for the same.

The rest of this paper is organized as follows. Section 2 discusses related work. Section 3 describes for MAV vision scenario the proposed target and measurement models as well as single and multiple target tracking. In section 4 we discuss at the multitarget filtering using the proposed models. Section 5 describes algorithm for maintaining target histories. Conclusions are drawn in section 6 along with discussions on other possible uses of the proposed model/algorithms and future work.

2. RELATED PRIOR WORK

Most existing work in vision-based navigation is based on ground moving vehicle, in a controlled environment. Ground vehicles have much less degrees of freedom of manoeuvre and assumptions about the environment make their travel easier.

Many authors have proposed use of feature points for vehicle pose estimation or autonomous navigation. While this has a rich theoretical background, too, we argue that optic flow is emerging as an important tool for many applications and is also useful for depth estimation, providing time to collision. It is very likely that the vehicle will be performing optic flow computation as a part of its execution cycle and therefore it makes sense not to consume additional resources in extracting feature points. Also, once features have been extracted, the objects need to be segmented by clustering the feature points for obstacle avoidance/target tracking, requiring more computation time.

Webb *et al*. [9] estimate the state of a flying vehicle from tracking motion of feature points in the scene using extended Kalman filter. The authors assume that the MAV is moving in a restricted area, and that a set of static feature points in that area are identified and fed to the algorithm in advance. The aim of our work is to identify objects for the purpose of tracking or avoiding them; these objects are not known a priori, and can appear and disappear at will. However, the authors have discussed important concepts that can be used in conjunction with our work to accomplish many tasks.

Souhila and Karim [10] have worked on optical-flow based robot obstacle avoidance. Their work is on ground-based vehicles. The authors have discussed important results for object depth estimation and we use their depth estimation model in our work.

Angelosante D. *et al*. [11] discuss the framework of finite set statistics for multiple target tracking. Their emphasis is on defining a reduced-complexity solution for multi-target state estimator.

Perhaps closest to our present work is that by Maggio E. *et al*. [12]. Authors approximate the target by the bounding rectangle (which can be considered as special case of a polygon used in our model), and propose a resampling strategy that improves the quality of Monte Carlo estimation from a tracking

perspective. The authors use State Dependent Variances (SDV) to account for the size of targets; it is assumed that objects far from the camera have smaller acceleration and size variations and hence the authors approximate scale and acceleration changes by random processes with standard deviations proportional to the object size. While this assumption may be true in some specific scenarios, it does not hold good in general, and certainly not in the scenario when MAV comes across objects of all possible sizes, shapes and speeds at many possible distances. Their work does not require them to consider 6-DoF movement model, either. Working with 6-DoF model makes the bounding rectangle of the same object at the same distance appear larger or smaller. However, the authors have made significant contributions that can be used with our model to improve estimation quality during implementation.

Mahler [4] provides an excellent coverage on random set theory, finite set statistics and filter approximations to multitarget filtering etc.; this work forms the basis for most models used in this paper.

3. SINGLE TARGET FILTERING

For the purpose of illustrating our model, we first consider the case where a single target is present in the field of view and is to be tracked. We describe the proposed state-transition and the measurement (observation) models for the single target case and discuss the filtering process i.e. the canonical (Kalman filter) predict-update cycle. The model is extended to multiple target scenario in section 4.

3.1. Sensor State Representation

The sensor state (Fig. 1) is represented as a vector $\hat{\boldsymbol{y}} = [x, y, z, \omega, \psi, \Phi, v_x, v_y, v_z, v_\omega, v_\psi, v_\Phi]^T$. Here, (x, y, z) are the three inertial position states corresponding to the north, east, down inertial coordinate system, (ω, ψ, Φ) the three Euler angles corresponding to the standard roll, pitch and yaw angles, (v_x, v_y, v_z) are the three velocity states corresponding to the body-fixed velocity vector and (v_ω, v_ψ, v_Φ) are the three angular velocity states corresponding to the total velocity, sideslip angle and angle of attack.

We do not assume that these parameters are known, nor do we attempt to estimate them in absolute terms; that is out of scope of our present work. We consider object motions relative to the sensor motion to be able to track objects. However, given appropriate calibration and mid-course correction, it is possible to use this model to estimate these parameters for the sensor and hence for the vehicle.

3.2. Target State Representation

Target is any object in the field of view. We do not consider point targets; vision allows us to consider its size, shape, colour, texture etc. However, we consider the target as sensed from the optic flow generated by it, approximated with a bounding polygon (Fig. 2). A target is represented by the following parameters.

– Corners of bounding polygon (Max 'S'): Each corner comprises of the following: $\text{corner}_i = [x_i, y_i, z_i, h_i, t_i, i_i]^T$; ($x_i$, y_i, z_i) are the coordinates of the corner and (h_i, t_i, i_i) are visual parameters representing hue, texture and intensity in a small area around the i^{th} corner.

– Velocity: Target's linear and angular velocities given by the vector velocity = $[\gamma_x, \gamma_y, \gamma_z, \gamma_\omega, \gamma_\psi, \gamma_\Phi]^T$ relative to the sensor

The target state vector $\hat{u}$ is given as follows.

$\hat{u} = [\text{corner}_1, \ldots \text{corner}_S, \text{velocity}]^T$

If number of corner points is less than max, remaining ones are filled with zeros. The limit on number of corner points decides how fine or coarse the representation will be.

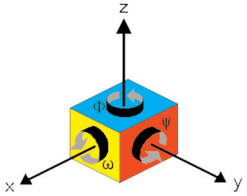

Figure 1. The 6-DoF Model

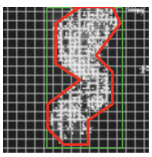

Figure 2. Bounding polygon and rectangle from optic flow representing a target

In many situations, it is important to maintain a consistent identity of objects over time. However, as we shall see in section 4, random sets are essentially that – sets, without any sequencing. We discuss a way of assigning and maintaining target IDs in section 5. With this, the procedure additionally stores the following parameters.

– Number of corners in the bounding polygon (max 'S')

– Corners of bounding rectangle: Represented as vectors $[x_1,y_1,z_1,h_1,t_1,i_1]^T \ldots [x_4,y_4,z_4,h_4,t_4,i_4]^T$

– Target ID (Filled in by data association procedure, described in a later section)

The locations are assumed to be relative to the origin of the field of view.

3.3. State-transition Model

The target state vector consists of number of corners, corner parameters as discussed in the previous section, and the velocity parameters. Rigid body motion is assumed; targets can move with respect to the three axes and rotate in the three planes, but shear and scaling operations are not allowed (changes in target size are due only to changes in distance between target and the vehicle).

If we knew in advance the shape of the target in 3D, the state-transition model would be given by the matrix equations as follows.

Consider (c_ω,s_ω), (c_ψ,s_ψ), (c_Φ,s_Φ) to be the cosines and sines of the three rotation angles. Then the rotation matrix is specified as

$$R = \begin{bmatrix} r_1 & r_2 & r_3 \\ r_4 & r_5 & r_6 \\ r_7 & r_8 & r_9 \end{bmatrix},$$

with

$$\begin{aligned} &r_1 = c_\omega * c_\Phi + s_\omega * s_\psi * s_\Phi, \quad r_2 = -c_\psi * s_\omega, \\ &r_3 = -c_\omega * s_\Phi + c_\Phi * s_\omega * s_\psi, \quad r_4 = s_\omega * c_\Phi - c_\omega * s_\psi * s_\Phi, \\ &r_5 = c_\omega * c_\psi, \quad r_6 = -s_\omega * s_\Phi - c_\omega * s_\psi * c_\Phi \\ &r_7 = c_\omega * s_\Phi, \quad r_8 = s_\psi, \quad r_9 = c_\psi * c_\Phi \end{aligned}$$

Consider

$$\tilde{T} = \begin{bmatrix} R & 0_3 \\ 0_3 & I_3 \end{bmatrix}, \quad \hat{T} = \begin{bmatrix} \tau I_3 & 0_3 \\ 0_3 & 0_3 \end{bmatrix},$$

Where τ: time elapsed between 2 successive computations (frame interval for computing optic flow); I_n: nxn Identity matrix; 0_n: nxn Zero matrix

The final state-transition model is given by

$$T = \begin{bmatrix} \tilde{T}_1 & 0_6 & \ldots & \ldots & \hat{T}_1 \\ 0_6 & \tilde{T}_2 & \ldots & \ldots & \hat{T}_2 \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ \ldots & \ldots & \ldots & \tilde{T}_S & \hat{T}_S \\ 0_6 & 0_6 & \ldots & 0_6 & I_6 \end{bmatrix}$$

Given number of corner points $n \leq S$, the $\tilde{T}$ and $\hat{T}$ matrices for the corners from n + 1 to S can be assigned values as 0_6.

In reality, however, we never know the real shape of the object, only of its projection on the sensor. This fact is crucial because when the object rotates in the YZ/XZ plane, not just the shape of its projection on the 2-D plane appears different, with possibly different number of corners, but the new corners may appear in between the original ones, making correspondence difficult (Fig 3).

An associated problem is that while the rigid body movement ensures all corners undergo the same transformation, it cannot be said for the bounding rectangle. The centroid also may get altered.

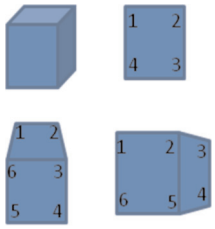

Figure 3. A 3D object with different 2D projections and corner sequencing

These issues could be addressed by making the model non-linear and complex, but it would have substantial computational time penalty. As a result we retain the above described linear model and handle problem of corner associations in data association step.

3.4. State-transition Density

We assume that the target motion follows a linear Gaussian model. The density is given by:

$f_{klk-1}(\hat{u}^k|\hat{u}^{k-1}) = N(\hat{u}^k; T^{k-1}\hat{u}^{k-1}; Q^{k-1})$ i.e. Normal density evaluated at $\hat{u}^k$, centered at $T^{k-1}\hat{u}^{k-1}$. Noise is assumed to be zero-mean. The noise covariance matrix Q is a diagonal matrix (it's realistic to assume that movements in all the directions are independent of each other); its entries are initialized to have large values; during filtering, the matrix is updated at each time step. $\hat{u}^k$ is the vector $\hat{u}$ at time k.

3.5. Measurement Model

Target measurement is carried out from the 3-D world's projection on the 2-D image plane followed by use of 2 frames at frame interval Ï (time elapsed between the two being τ) to compute optic flow fields. Perspective projection requires division by the z-coordinate which, in general, cannot be easily estimated using only one vision sensor. However, use of optic flow has been suggested in literature for obtaining depth perception, e.g. by Souhila K. and Karim [10]. We assume use of their model to obtain depth perception. The measurement vector is given below.

$$[x_1, y_1, 1, h_1, t_1, i_1, \ldots, x_s, y_s, 1, h_s, t_s, i_s, M_x, M_y, M_z, M_\omega, \gamma_\psi, \gamma_\Phi]$$

Where M_x, M_y, M_z are x, y and z translational components of optic flow respectively; and M_ω is rotational component of optic flow in XY plane. The z-coordinate cannot be measured; however, the depth of the object plays a crucial role and is considered for calculating the projection of the target on the image plane. We do not attempt to measure γ_ψ and γ_Φ; these values are filled randomly.

Having separated out the objects and their bounding polygons (Fig. 2), target state measurement is done using the following matrix equations.

Projection matrix (projection on image plane):

$$P = \begin{bmatrix} 1 & 0 & -x_c / z_c \\ 0 & 1 & -y_c / z_c \\ 0 & 0 & -1 / z_c \end{bmatrix} \text{ for each target corner}$$

Matrix for visual parameters:

$$V = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \text{ (read from the image)}$$

From these, we get projection for one corner as follows.

$$\tilde{M} = \begin{bmatrix} P & 0_3 \\ 0_3 & V \end{bmatrix}$$

Linear velocities using optic flow fields

$$L = \begin{bmatrix} C_x * \tau & 0 & 0 \\ 0 & C_y * \tau & 0 \\ 0 & 0 & C_z * \tau \end{bmatrix}$$ where

C_x, C_y: constants that convert velocity from meters/second to pixels/second using depth estimate; C_z: constant that estimates depth from translational component. We assume use of model suggested by [10] for depth estimate.

Angular velocity in XY-plane using optic flow

$$A = \begin{bmatrix} C_\omega * \tau & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$ where

C_ω: converts angular velocity from radians/second to pixels/second using depth estimate. It is difficult to compute the angular velocities in the YZ and XZ planes, and is also not required in our method; hence we do not consider them.

The velocity matrix is

$$\hat{M} = \begin{bmatrix} L & 0_3 \\ 0_3 & A \end{bmatrix}$$

The final measurement model is given by the following matrix. This matrix multiplied with the target state vector gives the observed state. Since we cannot measure what we do not see, the measurement applies only to the visible corners.

$$M = \begin{bmatrix} \tilde{M}_1 & 0_6 & \dots & 0_6 \\ 0_6 & \tilde{M}_2 & & 0_6 \\ \dots & & \dots & \dots \\ & & \tilde{M}_S & 0_6 \\ 0_6 & 0_6 & \dots & \dots \; 0_6 \; \hat{M} \end{bmatrix}$$

3.6. Likelihood Function

Under the given assumptions and modelling conditions, the sensor model is linear Gaussian, and the likelihood function is given by

$g_k(\check{z}^k|\hat{u}^k) = \mathcal{N}((\check{z}^k; M^k\hat{u}^k; \Lambda^k)$, where Λ is the covariance matrix of the measurement noise, assumed independent of the system noise. Λ is a diagonal matrix. We select the measurement noise as follows: variance for coordinates = (frameInterval*3) pixels, variance for velocities = 0.3 meters/sec, variance for hue and intensity = 0.03, variance for texture = its original value multiplied by 0.03.

3.7. Tracking

The goal of tracking is to estimate the probability density function (pdf) $p_{k|k}(\hat{u}^k|\check{z}^{1:k})$ of the target being in state $\hat{u}k$, given all observations up to time k. The estimation is performed recursively in 2 steps viz. prediction and update. Prediction uses state transition model (Section 3.4) to obtain the prior pdf

$$p_{k|k-1}(\hat{u}^k \mid \breve{z}^{1:k-1}) = \int f_{k|k-1}(\hat{u}^k \mid \hat{u}) p_{k-1|k-1}(\hat{u} \mid \breve{z}^{1:k-1}) d\hat{u}$$

The update step applies observation model (Section 3.6) and uses the Bayes' rule to find the posterior

$$p_{k|k}(\hat{u}^k \mid \breve{z}^{1:k}) = \frac{g_k(\breve{z}^k \mid \hat{u}^k) p_{k|k-1}(\hat{u}^k \mid \breve{z}^{1:k-1})}{\int g_k(\breve{z}^k \mid \hat{u}) p_{k|k-1}(\hat{u} \mid \breve{z}^{1:k-1}) d\hat{u}}$$

For linear f and g and Gaussian noise, the recursion can be solved using standard Kalman filter. For generic approximations, Monte Carlo estimation techniques are followed, where the densities are approximated as sums of dirac δ functions or particles.

3.8. Computational Complexity

The computations stated here are quite intensive. Although literature suggests real-time implementation of robust optic flow computations as well as to segment objects from the optic flow, their application with the rest of the operations in MAV remains to be seen. The complexity of computing optic flow depends on the specific algorithm used and its implementation.

As regards our model, the matrix size can be fairly large depending on the maximum desired number of corner points S. However, that both state-transition and measurement matrices are highly structured sparse matrices, reducing the matrix-vector multiplication to a few multiplications and additions on the number of corner points. The complexity of filtering process will depend on the chosen pdf.

3.9. An Example

The matrix listing with a target having just three corners becomes somewhat lengthy to write and comprehend; we illustrate the case with a single point.

The state transition matrix becomes

$$\begin{bmatrix} R & 0_3 & \tau I_3 & 0_3 \\ 0_3 & I_3 & 0_3 & 0_3 \\ 0_3 & 0_3 & I_3 & 0_3 \\ 0_3 & 0_3 & 0_3 & I_3 \end{bmatrix}$$

Target state vector with one corner point is

$$[x_1, y_1, z_1, h_1, t_1, i_1, \gamma_x, \gamma_y, \gamma_z, \gamma_\omega, \gamma_\psi, \gamma_\Phi]^T$$

The parameters are in world coordinate system.

Multiplying, we see that the predicted velocities remain the same as in present state (constant velocity model) and the visual parameters remain unaltered (we cater to variations in imaging conditions by providing broader variance for these parameters).

We measure optic flow magnitudes in X and Y directions (M_x, M_y) and its rotational component in the XY plane (M_ω). Using the model discussed in [10], we obtain depth of the object from the sensor. This allows us to calculate x_c, y_c, z_c, and M_z. C_x, C_y, C_z and C_ω depend on the depth of the object, camera field-of-view and resolution. The measurement model for one corner point then becomes

$$\begin{bmatrix} 1 & 0 & -x_c/z_c & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -y_c/z_c & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1/z_c & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & C_x * \tau & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & C_y * \tau & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & C_z * \tau & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & C_\omega * \tau & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

This matrix multiplied by target state vector gives projections of (x,y,z) coordinates on the XY plane, keeps the visual parameters unaltered and gives velocities in terms of optic flow magnitudes.

4. GENERALIZATION TO MULTITARGET SETTING

The vision sensor (colour camera) mounted on the nose of an MAV provides a stream of video; optic flow is computed from this video with frames taken at interval Ï (time interval τ), objects are segmented from optic flow and stored in a structure called Spatial Optic Flow with Time Trace (SoftTrace) using algorithm outlined in Fig. 4. Note that in this case also, like in many multitarget tracking problems, two objects placed very close together and with the same optic flow magnitude may be treated as single object. However, it is easy to see that this does not affect the applications like target tracking or obstacle avoidance.

Creation of SoftTrace: Preliminaries
Perform segmentation on the optic flow field. Targets are segments with significant optic flow magnitudes, and are approximated with a bounding polygon. Number of corners in polygon is a trade-off between computational complexity and precision. Computations are as follows:
Initialize
Current corner x = x_{min} of all points on target (Could be more than 1 points with this X-coordinate) Current corner y = y_{min} of all the points with x_{min} as their X-coordinates on target. We call this point MinPoint for the object Check next point on boundary in clockwise direction Current angle = Angle between current corner and new point (angle measured w.r. to positive Y-axis representing 0°)
Extract Required Corners
For each subsequent point on the boundary in clockwise direction New angle = Angle between new point and current corner IF ((New angle − Current angle) > threshold) Add new corner, replace current corner (Threshold chosen depending on the desired max number of corner points 'S')
Extract Visual Parameters
For each corner - Check corresponding pixel in image frame - Select 16 pixels near the corner, inside the object - Compute average hue and intensity - Compute texture parameter
Get Bounding Rectangle
- (x_{min}, y_{min}) = (x_{min}, y_{min}) of polygon - (x_{max}, y_{max}) = (x_{max}, y_{max}) of polygon

Figure 4. Extracting and Storing State Vector

Let $\in$ be the region in the field of view of the camera. Let there be **n** targets in $\in$ at a given point in time **k**. At the sensor, these targets may result into **m** observations[1]; the order in which they appear being unknown. The targets and observations at time **k** can be represented as finite sets

$$X^k = \{x_1^k, x_2^k .....x_n^k\} \in F(X), Z^k = \{z_1^k, z_2^k, ..., z_m^k\} \in F(Z)$$

Here $F(X)$ and $F(Z)$ are the respective collections of all finite subsets of X and Z.

Note that, in the single target scenario, we used the notation û to indicate target state essentially to avoid mixing up with the x-coordinate. Here we switch to x_i as per the convention most followed. The use is generally clear from the context; the subscript also helps. For an excellent and detailed understanding of multi-target filtering, the reader is referred to [4].

4.1. Initialization

There can be unpredictable number and locations of targets with varying characteristics. We model this using a multitarget Poisson process with mean as the centre of field of view and covariance matrix covering the entire field of view. The corner points are initialized to (0,0) with large variance.

[1]Imperfect imaging conditions, closely spaced objects, occlusions, man-made objects with sharp colour boundaries can be few reasons for missed detections or false alarms.

$f_{0|0}(\{x_1, ..., x_n\}) = e^{-\lambda} \lambda^n . f(x_1). ... f(x_n)$, where λ is the Poisson cardinality parameter specifying the expected number of targets. For initialization we take λ to be 10. $\{x_1, ..., x_n\}$ is a Random Finite Set (RFS).

4.2. Target Birth and Death

The appearance of targets in the field of view is a dynamic event. Existing targets go out of site and new ones emerge as vehicle manoeuvres. The initial appearance of a target is called as the birth and the disappearance of a target beyond a certain time period is referred to as the death of the target. Spawning refers to one target splitting into two. For the purpose of tracking, missed detection amounts to target death; spawning is treated as a target birth.

The birth, death and spawning of the targets can happen at any place in the visual frame. The spontaneous birth of a target from a random location is modelled as multitarget Poisson process similar to the initial conditions, except that we take the Poisson parameter λ, expected number of newly born targets, to be 3.

4.3. Multitarget State Transition

Given the multitarget state X^k at time k, the multitarget state X^{k+1} at time $k + 1$ is given by

$$X^{k+1} = [\bigcup_{x_i \in X^k} L^{k+1|k}(x_i)] \cup [\bigcup_{x_i \in X^k} B^{k+1|k}(x_i)] \cup \Gamma_k, \text{ where}$$

$L_{k+1|k}(x_i)$: RFS of target x_i from time k that survived at time $k + 1$. Can be $\{x_i^{k+1}\}$ with probability of living $p_{L,k}(x_i)$ or ϕ with probability 1- $p_{L,k}(x_i)$.

$B_{k+1|k}(x_i)$: RFS of targets spawned at time k + 1 from a target with previous state x_i

Γ_k: RFS of targets born spontaneously at time $k + 1$

Given these sets and probabilities, the true multitarget Markov density $f(X^{k+1}|X^k)$ can be derived from the belief-mass function [4].

4.4. Multitarget Likelihood

The RFS measurement model caters to clutter as well as uncertainties in optic flow field; it allows missed detections and false alarms. The multitarget measurement $\boldsymbol{Z^{k+1}}$ at the sensor will be

$$Z^{k+1} = [\bigcup_{x_i \in X^{k+1}} \Gamma_{k+1}(x_i)] \cup C_{k+1}, \text{ where}$$

$\Gamma_{k+1}(x_i)$: RFS of measurements at time $k+1$ for a target with state x_i. Can be $\{x_i^{k+1}\}$ with probability of detection $p_{D,k}(x_i)$ or ϕ with probability 1- $p_{D,k}(x_i)$

$C_{k=1}$: RFS of false measurements

It is assumed that the RFSs constituting the above equation are independent of each other.

Given these sets and the probabilities, the true likelihood function can be derived from the belief-mass function [4].

4.5. Multitarget Filtering

The goal of multitarget tracking is to estimate the pdf $p_{k|k}(X^k|Z^{1:k})$ of the target set being in state X^k, given all observations up to time k. The estimation is performed recursively in 2 steps viz. prediction and update.

Prediction amounts to obtaining the prior pdf

$$f(X^k \mid Z^{1:k-1}) = \int f_{k|k-1}(X^k \mid X^{K-1}) p_{k-1|k-1}(X \mid Z^{1:k-1}) \mu(dX)$$

where μ is an appropriate reference measure on $F(X)$, the set of all finite subsets of X.
The update step applies uses the Bayes' rule to find the posterior

$$p_{k|k}(X^k \mid Z^{1:k}) = \frac{g_k(Z^k \mid X^k) p_{k|k-1}(X^k \mid Z^{1:k-1})}{\int g_k(Z^k \mid X) p_{k|k-1}(X \mid Z^{1:k-1}) \mu(dX)}$$

This problem has no closed form solution. Particle filtering is possible as an approximation; however, the number of particles required is exponentially related to the number of targets. Literature suggests propagating only the moments of the posterior rather than the posterior itself to bring down the computational complexity. One prime focus of our current work is to examine the minimum set of statistics that need to be propagated in our scenario to obtain robust prediction/ update at the least possible cost.

5. DATA ASSOCIATION

Knowing the identity of a target or tracking its movement may not be required for applications like obstacle avoidance; however, many other real-life applications require maintaining consistent identity of the objects, and even being able to lock on to them after a possible missed detection or temporary occlusion. Probabilistic techniques such as muti-hypothesis tracker [13] have been proposed for data association, but most of them are highly computation-intensive. Maggio E. *et al.* [12] have proposed a graph-based technique. The peculiarity of our problem demands an innovative solution. As discussed in section 3.3, the bounding polygon or the centroid may not undergo the same transformations as all of the corner points, However, depending on the direction of rotations, at least one corner of the rectangle will undergo roughly the same order of transformations. Since all objects are subjected to the same transformations, the relative changes can be traced. Considering only one point also saves computation time.

The first step is to select the desired corner point from the observation and the prediction sets. If change in roll was non-zero, we counter it with opposite rotation. Now for each target in the observation set, if heading change was positive, select left corner else select left. If change in pitch was positive, select bottom corner else select top. To assist in speeding up the operations, we modify Softtrace as depicted in Fig. 5, storing objects grid-wise.

Completing SoftTrace
To assist in target lookup for data association, we organize SoftTrace as follows
Store objects grid-wise
Field-of-view divided into rectangular grids; grid size decided based on expected variance on position variables Each object placed in the grid that contains its MinPoint
Some more history
Object ID is assigned as GridId-UniqueNumber pair 'Last Seen At' field added to cater to temporary occlusions, missed detections; expiry period for permanently removing it from the list
Update Targets Identified by Data Association
After each cycle of predict-update, for each identified target do the following - If target exists in the table, replace by new vector; update 'Last seen at' entry (may require grid repositioning) - If it does not exist, make new entry, update 'Last seen at'
Delete expired objects, predict stale ones
- If, at the end of the search, the 'Last seen at' entry of some object has expired, remove it from the structure - If the 'Last seen at' entry has not expired but is old, replace the vector by predicted values instead of updated values, increase variance for position and velocity fields marginally; this will ensure that if the target did unexpected manoeuvres while being occluded/ missed, it can be locked on to on being seen again - Probability distributions are stored in an auxiliary structure

Figure 5. Completing SoftTrace

Now we will have a set of polygon corner points on the two sets. Use distance measures such as Mahalanobis distance between observation and prediction to arrive at tentative association. To identify possible clashes, set thresholds. Clashes can be resolved by taking similar distance measures between different corner points of the polygon as well as matching the visual features. If none matches, it's a new target.

We add the field 'Last Seen At' to indicate how long the object has been missing for. At each predict-update cycle, the multitarget filter computes the updated state vector and probability distribution, which is the basis for next prediction. Objects already existing in SoftTrace are verified, as shown in Fig 5, against elements of the set given by the data association step and their 'Last Seen At' field is set accordingly. If object exists, it is replaced; if it does not, and the missing period is beyond certain expiry period, it is removed from the list. If the object is missing for a small interval of time, we consider it as possible missed detection or temporary occlusion. In this case, its state vector is replaced by the one predicted by the filter and its variance values for position and velocity components are marginally increased in each cycle. Until the target has been removed from the list, it is considered alive and is sent to the filter process for next cycle of prediction-update.

5. CONCLUSION AND DISCUSSIONS

The combination of optic flow based object segmentation and random set theory to track multiple targets is attractive, but requires many issues to be addressed before it can be turned into real-time implementation on micro air vehicles. Some of the challenges to overcome include computational complexity, selection of appropriate reference measure, and data association in presence of relative rotation of targets in the XZ and YZ planes, where target shape as it appears on the projected plane can change. With advancements in hardware technology, more power will be available on board the MAV; however, it may still not be enough to handle all of these calculations in real-time requiring approximate techniques to be developed. At present we are ignoring the computational time and focusing on the other challenges.

Like any other multitarget tracking method, this method is also subject to resolving targets with different signatures, or targets of similar signatures located far away from each other. Vision resolves targets better than many other sensors, but mix up cannot be avoided if many identical targets are moving in the field of view doing rapid manoeuvres.

6. REFERENCES

1. J. Weber, J. Malik S. Devdas and P. Michel. Robust Computation of Optic Flow in a Multi-scale Differential Framework. *Intl J Computer Vision*, Vol 14: pp 12-20, 1995.
2. C. Cassisa, V. Prinet, L. Shao, S. Simoens and C. L. Liu. Optic Flow Robust Estimation in a Hybrid Multi-resolution MRF Framework. *Proc IEEE Intl Conf Acoustics*, *Speech and Signal Processing*, pp 793-796, 2008
3. E. P. Simoncielli, E. H. Adelson and D. J. Heeger. Probability Distribution of Optic Flow. *Proc IEEE Conf on Computer Vision and Pattern Recognition*, pp 310-315, 1991
4. R. P. S. Mahler. Statistical Multisource-Multitarget Information Fusion. Norwood, MA, Artech House, 2007
5. Vo Ba-Ngu and Ma Wing-Kin. The Gaussian Mixture Probability Hypothesis Density Filter. *IEEE Tr Signal Processing*, Vol 54, No 11: pp 4019-4104, Nov 2006
6. Y. Wang, J. Wu, A. A. Kassim and Huang Wei-Min. Tracking a Variable Number of Human Groups in Video Using Probability Hypothesis Density. *Proc 18th Intl Conf on Pattern Recognition, ICPR 2006*, Vol 3: pp 1127-1130
7. D. Clark and S. Godsill. Group Target Tracking with the Gaussian Mixture Probability Hypothesis Density Filter. Proc 3rd Intl *Conf Intelligent Sensors, Sensor Networks and Information, ISSNIP 2007*, pp 149-154, 2007
8. S. Simo, A. Vehtari and J. Lampinen. Rao-Blackwellized Particle Filter for Multiple Target Tracking. *J Information Fusion*, Vol 8: 2006
9. T. P. Webb and R. J. Prazenica. Vision-based State Estimation for Autonomous Micro Air Vehicles. *J Guidance, Control, and Dynamics*, Vol 30, No 3: pp 816-826, 2007
10. K. Souhila and A. Karim. Optical Flow Based Robot Obstacle Avoidance. *Intl J Advanced Robotic Systems*, Vol 4, No 1: pp 13-16, 2007

11. D. Angelosante, E. Biglieri and M. Lops. Multiple Target Tracking using Random Sets. *Proc EUSIPCO 2008, 2008*
12. E. Maggio, M. Taj and A. Cavallaro. Efficient Multi-target Tracking using Random finite sets. *IEEE Tr Circuits and Systems for Video Technology*, Vol 18, Issue 8: pp 1016-1027, Aug 2008
13. D. Reid. An Algorithm for Tracking Multiple Targets. *IEEE Tr Automation and Control*, Vol 24, No 6: pp 843-854, 1979