6D UAV pose estimation for ship landing guidance

Tiago Ferreira
Institute for Systems and Robotics
Instituto Superior Técnico
Lisbon, Portugal
tiagovferreira@tecnico.ulisboa.pt

Alexandre Bernardino
Institute for Systems and Robotics
Instituto Superior Técnico
Lisbon, Portugal
alex@isr.tecnico.ulisboa.pt

Bruno Damas
Institute for Systems and Robotics
CINAV - Centro de Investigação Naval
Lisbon, Portugal
bdamas@isr.tecnico.ulisboa.pt

***Abstract*—Landing an Unmanned Aerial Vehicle (UAV) aboard a patrol boat is a challenging task due to the unpredictable ship movement, being its automation essential. Automated solutions rely on the UAV pose estimation, usually obtained from onboard sensors. Given the onboard sensors' limitations and the power consumption, we propose an off-board pose estimation method. By relying on RGB images captured from a camera at the ship deck, our method directly estimates the UAV pose with respect to the landing site, removing the dependency on any additional sensors. We propose a model-based pose tracking method with a Rao-Blackwellized Particle Filter (RBPF), that models the translational motion of the UAV approximating the translation by a set of hypotheses and a distinct rotational distribution for each translation hypothesis using a autoencoder network trained for our UAV model. This allows to reduce the sample search space from 6D to 3D. Furthermore, we propose a particle weighting process combining the contributions from the rotation likelihood distribution and a detector-based likelihood. The training of the neural networks and the validation of the proposed method is made on a graphically realistic simulator. The results show that our weighting process has benefits when compared to the baseline and other state-of-the-art approaches. Furthermore, our approach successfully handles objects with geometric symmetries.**

***Index Terms*—Autonomous UAV landing, pose tracking, YOLO, symmetries, visual servoing**

I. Introduction

Portugal has an Exclusive Economic Zone (EEZ) whose area is eighteen times greater than its land territory. In this area, the country has the right to explore the natural resources and exercises its jurisdiction [1]. Moreover, it is defined a Search and Rescue Region (SSR), three times greater than the Portuguese EEZ, where the country should perform Search and Rescue (SAR) operations. The patrolling of such a vast area, currently executed using patrol boats, can be handled using UAVs, allowing to save energy and human resources.

One of the most critical UAV operations during an offshore mission is its landing on the retention system at the patrol boat, shown in Figure 1. Given the ship motion and oscillations, performing the UAV landing procedure is a challenging task. Landing piloting errors represent the most common human-caused UAV accident [2], and thereby, this procedure automation is required.

Automated landing control systems rely on the UAV pose estimation, commonly obtained from onboard sensors. However, besides the power consumption, there are several limitations regarding onboard sensors, such as the error propagation of inertial sensors and the signal jamming of the Global Navigation Satellite System (GNSS). Thereby, in this article, we propose an off-board pose estimation method relying only on RGB images captured from a camera located at the ship deck. This allows us to directly estimate the pose regarding the landing site, removing the dependency from any additional onboard sensors. Furthermore, by performing the pose tracking from the ship, we can use higher computational power and better cameras.

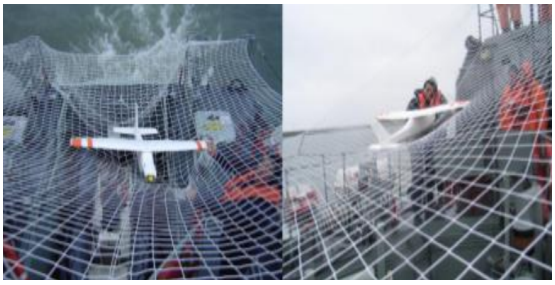


Fig. 1: The UAV retention system at the landing site. This image was adapted from [25].

Our approach relies on the UAV geometrical model to perform the pose estimation. However, the UAV geometrical shape has different symmetries, as shown in Figure 2, and the projection of different poses can produce images with similar appearances. Furthermore, the UAV landing procedure can occur at different day times and with different atmospheric conditions. Therefore, our method should be robust to symmetries, different illumination conditions, and different background scenarios.

Fig. 2: The UAV under symmetrical rotations. While the UAV on the left image moves towards the camera, the UAV on the right image moves away from the camera.

The method presented in this article is based on the approach proposed by Deng *et al* [3]. The baseline method estimates object poses by performing a Rao-Blackwellized Particle Filter (RBPF) [4] iteration for each new frame. The translation posterior distribution is represented by a set

of hypotheses, being the rotation distribution dependent on those hypotheses. The rotation distribution is represented by discretizing the Euler rotation space over 5 degree-sized bins, allowing to track multiple plausible rotation hypotheses simultaneously. The method observes the rotation distribution by encoding the UAV appearance in the input image and comparing it with pre-computed encodings for each discretized rotation. In this way, given that rotations with similar appearance have similar encodings, the observation outputs high density for each plausible rotation. Our innovation regards the particle weighting process. The baseline method calculate the particle weights by marginalizing the rotation distribution. In alternative, we estimate the weights by combining the rotation marginalization with a likelihood distribution based on the YOLO [6] object detection network. By not applying any translation observation at a regular iteration, the baseline method can't recover when there is a tracking loss, being in an open loop with respect to the object position. Each of the components of the weighting process gives a different contribution to the overall result.

The main contributions of this work are:

- A new particle weighting process combining detection and the rotation distribution;
- The study of the contribution of each weight component to the overall result;
- State-of-the-art results when comparing with the previous approach, from [5].

The remaining of this article has the following structure. In Section II, we overview the works related to our problem. Afterward, in Section III, we summarize the proposed approach. Section IV provides the experimental results on different datasets. Finally, in Section V, we provide the concluding remarks and discuss directions for future research.

II. Related Work

In this section, we will address the related work focusing on the UAV pose estimation (Sub-section II-A), object pose detection (Sub-section II-B), and object pose tracking (Sub-section II-C).

A. UAV pose estimation

Most UAV pose estimation techniques rely on cameras and sensors aboard the vehicle, although this further constrains the flight autonomy. Other works propose ground-based pose estimation techniques. Many ground-based techniques, such as [7], rely on markers aboard the vehicle. The markers are usually emitters/reflectors on the Infrared (IR) spectrum, whose light is captured by IR cameras. Other techniques are deployed using RGB cameras and rely on object features or region-based statistical models of the foreground and background. One example is the method of Jin *et al* [8], who estimates a quadcopter pose by performing a Perspective-n-Point (PnP) algorithm over keypoints regressed from a relational graph network. The pose estimation accuracy is usually constrained by the UAV distance to the camera, particularly when using monocular camera configurations. This issue can be mitigated by applying a stereo or multiple-camera framework. Rozantsev *et al* [9] estimates a quadcopter pose using images captured from multiple RGB cameras. The authors propose a new Bundle Adjustment (BA) formulation regularized by the UAV flight dynamics model. Many approaches are deployed on some temporal filtering technique, whether it is a Kalman filter [7]–[9] or a particle filter technique [5]. On a previous work, [5] estimates a fixed-wing UAV pose from images captured by a monocular RGB camera, using a particle filter. The main contributions of [5] are the use of directional statistics in tracking and the creation of the Unscented Bingham-Gauss Filter (UBiGaF).

B. Object pose detection

Pose detection encloses the pose estimation methods that depend only on the current frame to perform the estimation. Many pose detection techniques, such as [10], are dependent on the depth information from RGBD cameras. Other techniques apply the depth information when available, using it to refine the initial pose estimation through the Iterative Closest Point (ICP) algorithm. However, RGBD cameras are not suitable to be applied in open environments, such as the one our approach is deployed on. Many pose detection methods obtain coarse pose estimations by performing template matching [10], [11]. Template matching techniques compare features from the input images with pre-computed features. Most of the time, the coarse pose estimations are then refined, either using the ICP [10]–[13] algorithm or by aligning the object edges [12]. Recently, deep learning-based methods have become predominant. Whereas some methods directly regress the pose hypotheses [12], others form a pipeline of different sub-networks to regress pose parameters [13], [14]. Alternatively, some Convolution Neural Networks (CNNs) architectures produce features, as keypoints, subsequently used to compute the pose estimation [15]. The most common limitation of these pose detection techniques is the defective handling of symmetrical rotations.

C. Object pose tracking

Pose tracking comprises the pose estimation techniques that use information obtained on the current and previous frames to perform the estimation. Most of the pose tracking approaches propose energy function optimization techniques. The energy function tries to approximate the appearance of the last estimated pose by the appearance of the actual frame. The object's appearance can be coded through different kinds of features. Tjaden *et al* [16] propose an energy function optimization technique relying on global color histograms to model the background and foreground. In alternative, Zhong *et al* [17] proposed an energy function relying on two kinds of features. While the object contour regions were modeled using local color histograms, the object interior was modeled by the color gradients. Zhong *et al* [18] combine energy function optimization with deep learning networks. By applying an autoencoder to produce foreground/background masks, the method gains robustness to object occlusions. Meanwhile,

Manhardt *et al* [19] proposes a deep learning method aiming to optimize the pose estimation by regressing pose updates. In contrast, some tracking techniques estimate the pose using temporal filtering techniques. One example is the approach of Deng *et al* [3], who estimates object poses using an RBPF framework. Its differentiating factor is estimating the rotation likelihood through an autoencoder, leading to greater robustness to symmetrical objects. However, by not applying any translation detector at a regular iteration, the method is susceptible to diverge in presence of non-linear movements.

III. Proposed approach

We base our method on the approach of Deng *et al* [3], that consists in a Rao-Blackwelised Particle Filter, for which we contribute with a novel likelihood function for particle weighing. This section begins with an overview of the Rao-Blackwellized Particle Filter (RBPF), the particle composition, and the rotation distribution representation. Then, in Sub-section III-B, we define the rotation distribution. Sub-section III-B2 describes the translation observation technique. Finally, Sub-section III-D describes the particle filter processing.

A. RBPF overview

The Rao-Blackwellized Particle Filter (RBPF) is an efficient particle filtering technique that divides the state space into two groups of variables. In our case, we divide the state space in the translation state variables, T, and the rotation state variables, R. Under this decomposition, we have a posterior probability distribution of the type $P(T_{0:k}, R_{0:k}|Z_{0:k})$, with $Z_{0:k}$ representing the set of observations until iteration k. The posterior probability distribution can be decomposed by using the chain rule,

$$P(T_{0:k}, R_{0:k}|Z_{0:k}) = P(R_{0:k}|T_{0:k}, Z_{0:k})\, P(T_{0:k}|Z_{0:k}), \quad (1)$$

with $P(R_{0:k}|T_{0:k}, Z_{0:k})$ representing the rotation probability distribution dependent on the translation and observation, and $P(T_{0:k}|Z_{0:k})$ being the posterior probability distribution of the translation.

The RBPF gains its efficiency from the assumption that the distribution $P(R_{0:k}|T_{0:k}, Z_{0:k})$ is analytically tractable. Using the mentioned assumption, only a subset of the state space is approximated by the particle filter, and the number of particles is reduced. Therefore, under the RBPF assumption, equation (1) becomes,

$$P(T_{0:k}, R_{0:k}|Z_{0:k}) \approx \sum_{i=1}^{N} w_k^i\, \sigma_{T_{0:k}^i}(T_{0:k})\, P(R_{0:k}|T_{0:k}^i, Z_{0:k}), \quad (2)$$

where N is the number of particles, w_k^i is the weight of the ith particle in iteration k, $\sigma_{T_{0:k}^i}(T_{0:k})$ is the Dirac delta located in $T_{0:k}^i$, and $P(R_{0:k}|T_{0:k}^i, Z_{0:k})$ is the rotation distribution dependent on the ith translation hypothesis and the set of observations. In this way, based on [3], we define the particles as

$$\chi_k = \left\{T_k^i, P(R_{0:k}|T_{0:k}^i, Z_{0:k}), w_k^i\right\}_{i=1}^{N} \quad (3)$$

and we represent the distribution $P(R_{0:k}|T_{0:k}^i, Z_{0:k})$ as a three-dimensional tensor in which the dimensions are the Euler angles a ordered as roll (α), pitch (β), and yaw (γ). To represent all rotations, the roll and yaw angles range in the interval $[-180°, 180°]$ and the pitch angle range in $[-90°, 90°]$. By discretizing the distribution over 5-degree bins, the resulting tensor has a dimension of $72 \times 37 \times 72$ bins.

B. Rotation distribution

As previously mentioned, we consider the distribution $P(R_{0:k}|T_{0:k}^i, Z_{0:k})$ to be analytically tractable. Using Bayes rule and the Markov assumption on state transitions, we can rewrite $P(R_{0:k}|T_{0:k}^i, Z_{0:k})$ as

$$P(R_{0:k}|T_{0:k}^i, Z_{0:k}) \propto P(Z_k|R_k, T_k^i)\, P(R_k|R_{k-1})\, P(R_{0:k-1}|T_{0:k-1}^i, Z_{0:k-1}), \quad (4)$$

where: $P(R_k|R_{k-1})$ is the rotation motion prior, defined in III-B1; $P(Z_k|R_k, T_k^i)$ is the observation likelihood distribution, defined in III-B2; and $P(R_{0:k-1}|T_{0:k-1}^i, Z_{0:k-1})$ is the rotation distribution from the previous iteration.

1) Rotation motion prior: Based on [3], we consider the rotation motion to be locally static. Thereby, the rotation motion prior is defined as

$$P(R_k|R_{k-1}) = \mathcal{N}(R_{k-1}, \Sigma_R), \quad (5)$$

where Σ_R is the rotation covariance matrix. As such, the rotation proposal distribution is given by the convolution of $P(R_{0:k-1}|T_{0:k-1}^i, Z_{0:k-1})$ with a Gaussian kernel K_R

$$P(R_k|R_{k-1})\, P(R_{0:k-1}|T_{0:k-1}^i, Z_{0:k-1}) = P(R_{0:k-1}|T_{0:k-1}^i, Z_{0:k-1}) * K_R. \quad (6)$$

2) Observation likelihood: For a given rotation, the UAV appearance can be represented by features obtained by diverse methods. In this article, we adopt the same solution as [3], representing object rotations using the encoding features obtained from a trained autoencoder.

As depicted in Figure 3, both the autoencoder input and output are 128×128 pixel RGB images. The network training objective is to reconstruct the UAV's appearance at the input image removing the background. Figure 4 exemplifies three network input, target, and output images. The encoding is a network by-product, obtained from the encoder sub-network. Being invariant to illumination and background changes, the network's encoding is used as a feature describing the UAV rotation. We synthesized the training images with the simulator of [5], which projects the UAV geometrical model on a background image. Given the considered deployment scenario of this work, we created a sea-related background database representing different day-times and different degrees of cloudiness. We rendered every training image under the same canonical translation,

$$T_o = [0, 0, z_o], \quad (7)$$

where $z_o = 10$ meters. In order to be used as input of the autoencoder, every training image was cropped under a Region of Interest (ROI) with a canonical length S_o. The canonical

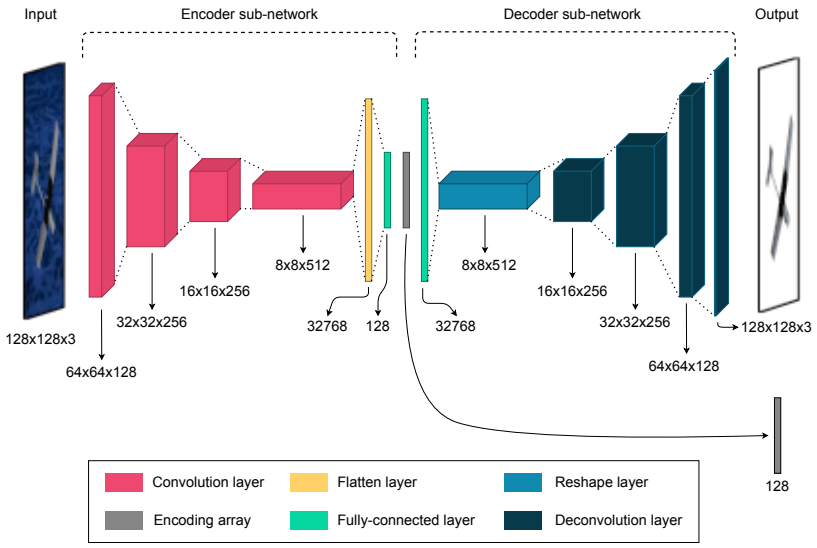
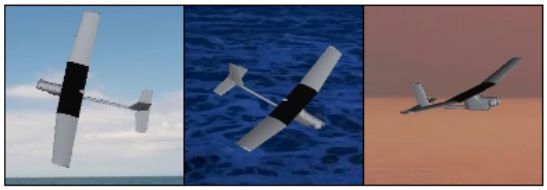
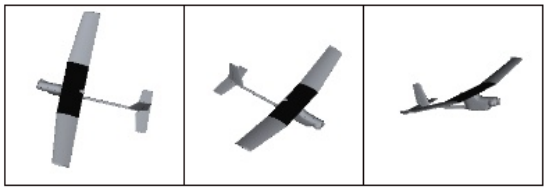


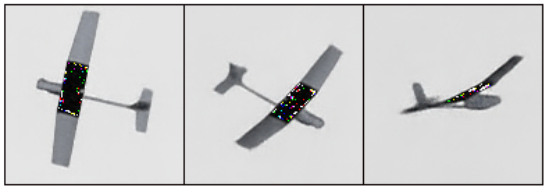
Fig. 3: The autoencoder's structure depiction. Bellow each block is defined its dimension.



(a) Autoencoder input.



(b) Autoencoder target.



(c) Autoencoder output.

Fig. 4: Correspondent examples of the autoencoder input, target, and output images.

length was defined such that the UAV projection of every rotation fits inside the canonical ROI [3].

Based on [3], the observation likelihood distribution is estimated by comparing the encoding of the UAV appearance on the input image with the set of encodings from the codebook. The codebook holds pre-calculated encodings obtained by processing synthetic images rendered using discretization of rotations described in III-A. To obtain the input image encoding, we begin by cropping an ROI. Given the translation hypothesis T_k^i, the ROI center coordinates are calculated as $u_k^i = f_x \frac{x_k^i}{z_k^i} + p_x$ and $v_k^i = f_y \frac{y_k^i}{z_k^i} + p_y$ where $[u_k^i, u_k^i]^T$ is the ROI's center coordinates, $[x_k^i, y_k^i, z_k^i]^T$ is the translation hypothesis T_k^i, and f_x, f_y, p_x, and p_y are the camera intrinsic parameters. The ROI dimensions are obtained from the scale factor

$$S_k^i = \frac{z_o \, S_o}{z_k^i}, \tag{8}$$

where S_k^i represents the ROI length. After rescaling the ROI it is used as an input of the encoder sub-network, producing the input image encoding.

Following [3], two encodings are compared by calculating their cosine similarity. Calculating the cosine similarity between the input image encoding and the codebook, we obtain a similarity tensor. The observation likelihood distribution is calculated as the normalization of the similarity tensor. By doing so, rotations with high similarity receive a high density, and the rotations with low similarities receive a density value near zero. Specifically, the observation likelihood distribution is calculated as

$$P(Z_k|T_k^i, R_k) = \mathcal{N}\left(CS_k^{max} - CS(Z_k|T_k^i, R_k), \sigma_{CS}^2\right), \tag{9}$$

where: $CS(\cdot)$ is the cosine similarity; σ_{CS} is the observation likelihood normalization standard deviation; and CS_k^{max} is the maximum value over all the cosine similarities evaluated at iteration k.

The observation likelihood is also sensible to the translation T_k^i. Erroneous translations hypotheses produce ROIs not centered in the UAV or with a wrong scale. Since the autoencoder was trained using a canonical ROI, the similarity tensor is sensible to translation errors.

C. Translation Observation

To obtain a likelihood distribution based on the UAV detection, a translation observation technique is needed. This

observation will be useful to the detection component of the particle weighting process, in III-D3. We base our translation observation on the parameters from ROIs estimated with the YOLO object detector [6]. The network was trained by [5] on a synthetic dataset with different poses and background images. The observed translation at iteration k can be defined as $T_k^Z = [x_k^Z, y_k^Z, z_k^Z]$. From detected ROI center we estimate x_k^Z and y_k^Z. Given that the UAV rotation influences the detected ROI, we estimate z_k^Z using a rotation-dependent canonical ROI length. The canonical ROI lengths were pre-calculated by running the YOLO detection on each image of the codebook dataset. In this way, equation (8) is adapted to

$$z_k^Z = \frac{z_o \cdot S_o^R(\tilde{R})}{\max(w, h)}, \tag{10}$$

where w and h are respectively the ROI width and height, $S_o^R(\tilde{R})$ is the canonical length of rotation $\tilde{R}$, and $\tilde{R}$ is the rotation with the highest density value from $P(Z_k|T_k^i, R_k)$.

D. Filtering

We divided the filtering processing into five sub-processes. In III-D1, we define the initialization of the particle filter at the first iteration. Afterward, III-D2 defines the processing of a regular iteration. Our novel weighting technique is detailed in III-D3. Then, we describe the resampling technique and the re-initialization criterion in III-D4. Finally, in III-D5, we present how the pose is estimated from the current set of particles.

1) Filter initialization: The filter initialization estimates a prior distribution based only on the observation. This initial iteration can be thought of as a pose detection iteration since there is not any previous information to perform tracking.

Our filter initialization begins by detecting an ROI using the YOLO detector [6]. The initial rotation distribution $P(R_0|T_0, Z_0)$ is estimated as the initial observation likelihood distribution, obtained using the detected ROI as the autoencoder input. After calculating the rotation distribution, the translation hypotheses are estimated using the previously defined translation observation technique with different values of $\tilde{R}$. Instead of estimating $\tilde{R}$ as the rotation with the maximum value of $P(Z_0|T_0, R_0)$, we estimate the set of rotations corresponding to the N higher values of the distribution. In this way, we have some variability to the initial translation hypotheses.

2) Filter iteration: After the initialization, for every new frame, a filter iteration is performed. Figure 5 depicts the procedure of one filter iteration.

The filter begins by sampling from the translation proposal distribution. As [3], we consider the translation to follow a constant velocity model. At iteration k, we sample the translation hypotheses from

$$T_k^i \sim \mathcal{N}\left(\hat{T}_k^i, \Sigma_T\right), \tag{11}$$

where $\hat{T}_k^i$ represents the translation obtained by applying the motion model on the ith particle, and Σ_T is the translation covariance matrix. Given the locally linear model assumption, $\hat{T}_k^i$ is defined as

$$\hat{T}_k^i = \begin{cases} T_{k-1}^i & \text{if } k = 1 \\ T_{k-1}^i + \alpha\left(T_{k-1}^i - T_{k-2}^i\right) & \text{if } k > 1 \end{cases}, \tag{12}$$

where α represents a velocity weighting factor [3]. Given the translation hypotheses sampled from the translation proposal distribution, the next step is to estimate the rotation distribution through equation (4).

3) Proposed Weighting Strategy: Once we have the translation hypotheses and the rotation distribution, the particle weights estimation is performed. The particle weights are estimated by calculating the likelihood distribution $P(Z_k|T_{0:k}^i, Z_{0:k-1})$. As previously discussed, our main contribution is a weighting process combining a detection-based likelihood with the autoencoder-based likelihood proposed by [3]. We begin by assuming that the observation, Z_k, is composed of two sub-observations, such that

$$Z_k = \left\{Z_k^A, Z_k^Y\right\}, \tag{13}$$

with Z_k^A representing the observation obtained from the autoencoder and Z_k^Y representing the observation obtained using the YOLO object detector [6]. Its joint density distribution is given by

$$P(Z_k^A, Z_k^Y|T_{0:k}^i, Z_{0:k-1}) = P(Z_k^Y|T_k^i, Z_k^A)\, P(Z_k^A|T_{0:k}^i, Z_{0:k-1}), \tag{14}$$

where $P(Z_k^Y|T_k^i, Z_k^A)$ is the YOLO-based likelihood distribution assumed independent of the past, and $P(Z_k^A|T_{0:k}^i, Z_{0:k-1})$ is the autoencoder-based likelihood distribution.

The YOLO-based likelihood distribution per particle is defined as

$$P(Z_k^Y|T_k^i, Z_k^A) = \mathcal{N}\left(T_k^Z - T_k^i, \Sigma_k^Z\right), \tag{15}$$

where T_k^Z is the previously defined translation observation and Σ_k^Z is the translation covariance matrix. As the translation observation error was observed to be linearly dependent on the UAV distance from the camera, we defined Σ_k^Z dependent on the last z-coordinate value.

Following [3], the autoencoder-based likelihood distribution is given by

$$P(Z_k^A|T_{0:k}^i, Z_{0:k-1}) = \textstyle\sum_{R_k} P(Z_k^A|T_k^i, R_k)\, P(R_k|T_{0:k}^i, Z_{0:k-1}), \tag{16}$$

where $P(Z_k^A|T_k^i, R_k)$ is the observation likelihood distribution defined in equation (15), and $P(R_k|T_{0:k}^i, Z_{0:k-1})$ is a rotation distribution given by

$$P(R_k|T_{0:k}^i, Z_{0:k-1}) = \textstyle\sum_{R_{k-1}} P(R_k|R_{k-1})\, P(R_{k-1}|T_{0:k-1}, Z_{0:k-1}). \tag{17}$$

4) Resampling and re-initialization criterion: Afterward, the particles are resampled according to their importance weights. As in [3] we resample the estimated particles using the systematic resampling method [20] and a filter re-initialization criterion is verified. If the value of CS_k^{max}, introduced in the equation (9), is lower than a threshold for five consecutive iterations, the filter is re-initialized.

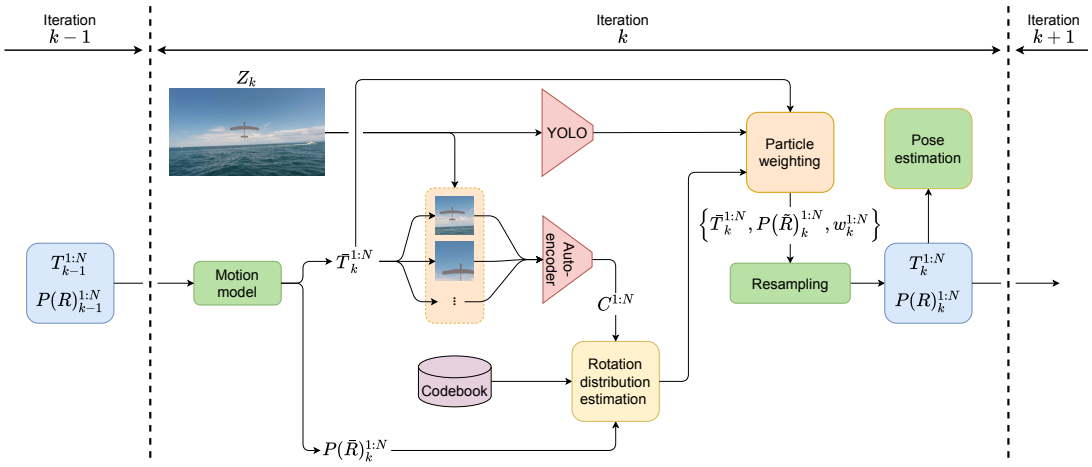


Fig. 5: The proposal's architecture diagram. To simplify the depiction we represent the rotation distributions as $P(R)_k^{1:N}$. This diagram was adapted from [3].

5) Pose estimation: After each iteration, given the resampled set of particles, the pose is estimated. The translation is simply estimated as the weighted average over all the translation hypotheses. In order to calculate the rotation, we begin by marginalizing the translation component from distribution $P(R_{0:k}|T_{0:k}^i, Z_{0:k})$, resulting in

$$P(R_{0:k}|Z_{0:k}) = \sum_{i=1}^{N} w_k^i \, P(R_{0:k}|T_{0:k}^i, Z_{0:k}), \tag{18}$$

with $P(R_{0:k}|Z_{0:k})$ representing the rotation posterior probability distribution. The rotation is estimated as a weighted average over all the discretized rotations, weighted by the values of $P(R_{0:k}|Z_{0:k})$. Given the Euler angles limitations, the average is calculated in the quaternion space. As [3], we calculate the quaternion weighted average following the approach of Markley *et al* [21].

IV. Experiments and Results

In this section, we evaluate the proposed approach. We begin by describing some implementation details such as the network's training and the filter parameter values. Afterward, in Sub-sections IV-B and IV-C, respectively, we introduce the datasets and metrics we use to assess the obtained results. Then, in section IV-D we evaluate the results of the full system. Moreover, in Sub-section IV-F, we evaluate and discuss the contribution of each weight component. Finally, in Sub-section IV-E, we assess the observation likelihood distribution performance regarding symmetrical rotations.

A. Implementation details

For training the autoencoder network, we rendered a dataset composed of 300 000 synthetic images. The images were rendered by projecting the UAV geometrical model under different rotations, uniformly sampled from the rotation space, on different sea-related background images. We rendered the training images using the simulator of [5]. Figure 6 shows some autoencoder training images examples. We trained the network using as loss function the Mean Squared Error (MSE), and the optimization function was Adam [22]. The Adam optimization function hyper-parameters were defined as $\alpha = 0.0002$, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-7}$. The network was trained for 16 epochs. The weights of each network layer were initialized using the Xavier uniform initializer [23]. To perform the YOLO object detector, we use the weights trained by [5]. The network was trained using a dataset composed of 335 767 images with different poses and background images. Our approach is evaluated by performing the particle filter applying the parameters defined in Table I. The dynamical standard deviations from Σ_T^Z were empirically defined as

$$\begin{cases} \sigma_{Dyn_x} = 0.025100 \times z_{k-1} - 0.085096 \\ \sigma_{Dyn_y} = 0.015304 \times z_{k-1} - 0.049382 \\ \sigma_{Dyn_z} = 0.113718 \times z_{k-1} - 0.172764 \end{cases}, \tag{19}$$

where z_{k-1} is the estimated pose z-coordinate of the last iteration.

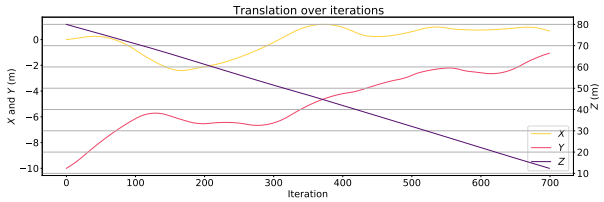
Fig. 6: Autoencoder training images examples.

TABLE I: Filter parameters definition.

Symbol	Name	Value
Σ_T	Translation motion prior covariance	$\begin{bmatrix} 0.07^2 & 0 & 0 \\ 0 & 0.07^2 & 0 \\ 0 & 0 & 0.7^2 \end{bmatrix}$ (m^2)
α	Velocity weighting factor	0.7
Σ_R	Rotation motion prior covariance	$\begin{bmatrix} 5^2 & 0 & 0 \\ 0 & 5^2 & 0 \\ 0 & 0 & 5^2 \end{bmatrix}$ (degrees^2)
Σ_T^Z	Translation likelihood distribution covariance	$\begin{bmatrix} \sigma^2_{Dyn_x} & 0 & 0 \\ 0 & \sigma^2_{Dyn_y} & 0 \\ 0 & 0 & \sigma^2_{Dyn_z} \end{bmatrix}$ (m^2)
σ_{CS}	Rotation likelihood normalization standard deviation	0.03
λ_{CS}	Cosine similarity threshold	0.91

B. Datasets

We evaluate our approach using two datasets, the test dataset of [5], and a dataset rendered with a linear translation motion, TransRot. [5] test the approach on a synthetic video simulating a UAV landing procedure on the patrol boat. The UAV follows a non-linear trajectory for both the translation and rotation. Figure 7 shows the UAV translation and rotation under [5] dataset. In order to evaluate the contributions of each weight component, we rendered the TransRot dataset. The TransRot dataset combines a linear translation with a non-linear rotation movement. Figure 8 shows the UAV translation and rotation under the TransRot dataset.



(a) Ground-truth translation.

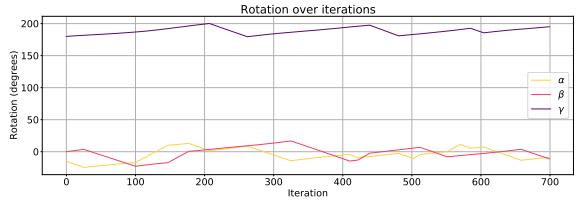
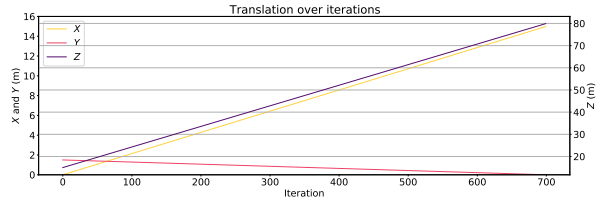


Fig. 7: Ground-truth pose on the dataset of [5].

C. Metrics

In this article, we evaluate the results using the same metrics as [5]. The translation error is evaluated using the Euclidean



(a) Ground-truth translation.

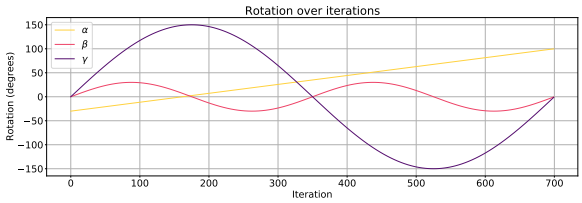


Fig. 8: Ground-truth pose on the TransRot dataset.

Distance (ED), defined as

$$ED_k = ||T_k - \tilde{T}_k||, \tag{20}$$

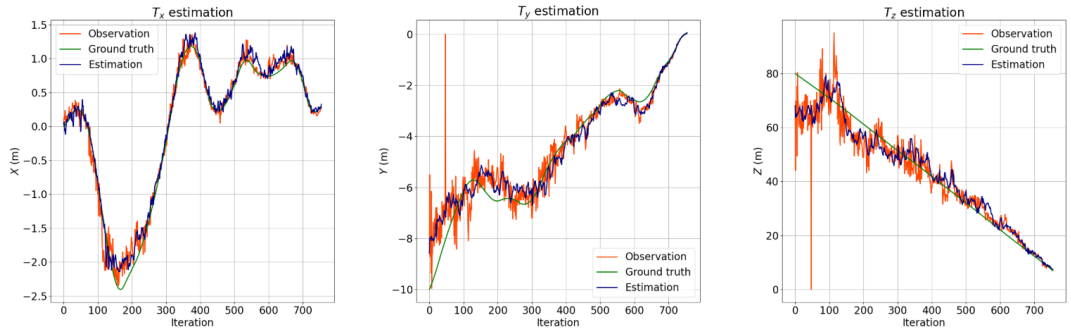
where T_k is the ground truth translation in iteration k, and $\tilde{T}_k$ is translation estimation. The rotation error is evaluated using

$$Error_k^{Rot} = \frac{180}{\pi}\sqrt{\frac{\left\|\log_m\left(R_k^T\,\tilde{R}_k\right)\right\|_F^2}{2}}, \tag{21}$$

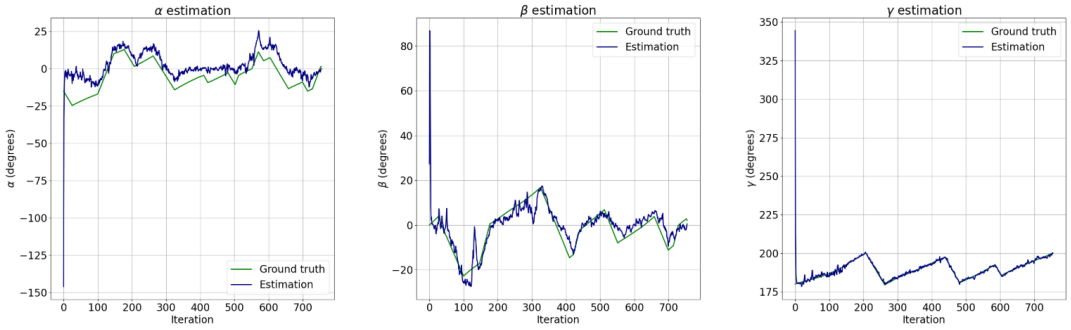
where $\log_m$ is the matrix logarithm, $\|\cdot\|_F$ is the Frobenius norm, R_k is the ground truth rotation matrix in iteration k, and $\tilde{R}_k$ is the estimated rotation matrix [5]. To summarize the obtained errors from equations (20) and (21) over all the iterations, we compute their average and standard deviation.

D. Full system performance

We evaluate the full system performance by performing our approach in Santos *et al* [5] dataset. Figure 9 shows the pose estimation results as well as the ground truth trajectory and the raw translation observations. A video showing our approach estimation results can found in https://www.youtube.com/watch?v=hTP0ikUciXc. We can observe that our approach can successfully track a trajectory simulating a landing procedure. Moreover, from these results, we can observe that our method is robust to non-linear trajectories despite that the assumed motion models are linear. Table II compares the numerical results of our approach with [5]. We can observe a great improvement in the UAV translation estimation. In particular, the median translation error is reduced by 46%, and the 95% percentile is reduced by 38%. The rotation estimation gets small improvements regarding the results of [5]. Although we have obtained higher errors in the 5% and 25% percentiles, we have lower errors in the remaining quartiles. In particular, we observe a reduction of 26% on the 95% percentile error. The error in the lower quartiles is justifiable by the size of the rotation discretized bins. Although the reduction of the size of the bins would reduce the rotation error, the computation time would increase substantially.



(a) Translation estimation.



(b) Rotation estimation.

Fig. 9: Translation and rotation estimation in Santos *et al* [5] dataset.

TABLE II: Comparison the results of our method with Santos *et al* [5] approach in Santos dataset.

Pose component		5% Percentile	25% Percentile	Median	75% Percentile	95% Percentile
Translation (m)	Our approach	**0.26**	**1.10**	**2.55**	**5.13**	**10.83**
	Santos *et al* [5]	0.29	1.28	4.73	8.87	17.44
Rotation (degrees)	Our approach	3.29	6.07	**7.89**	**10.10**	**17.60**
	Santos *et al* [5]	**1.25**	**4.46**	8.63	14.03	23.88

E. Rotation ambiguities handling

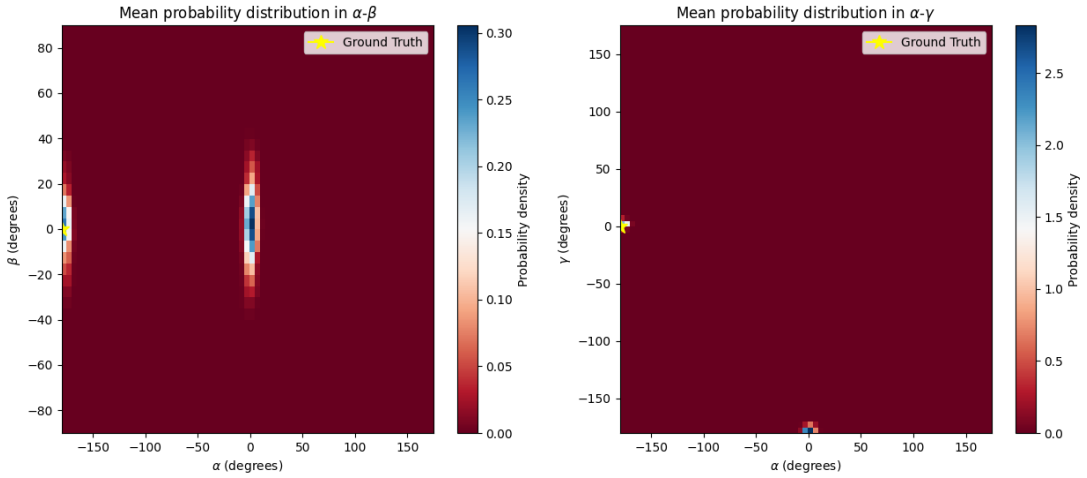
In this sub-section, we evaluate the observation likelihood distribution regarding the symmetrical rotations observed in Figure 2. Figure 10 shows the observation likelihood distributions obtained for the symmetrical rotations represented in Figure 2. As expected, the resulting likelihood distributions are very similar. We can observe that both likelihoods have two local maxima corresponding to the symmetrical rotations. Thereby, we can conclude that given an ambiguous UAV projection, the observation likelihood distribution observes all the plausible rotations.

F. Weight components evaluation

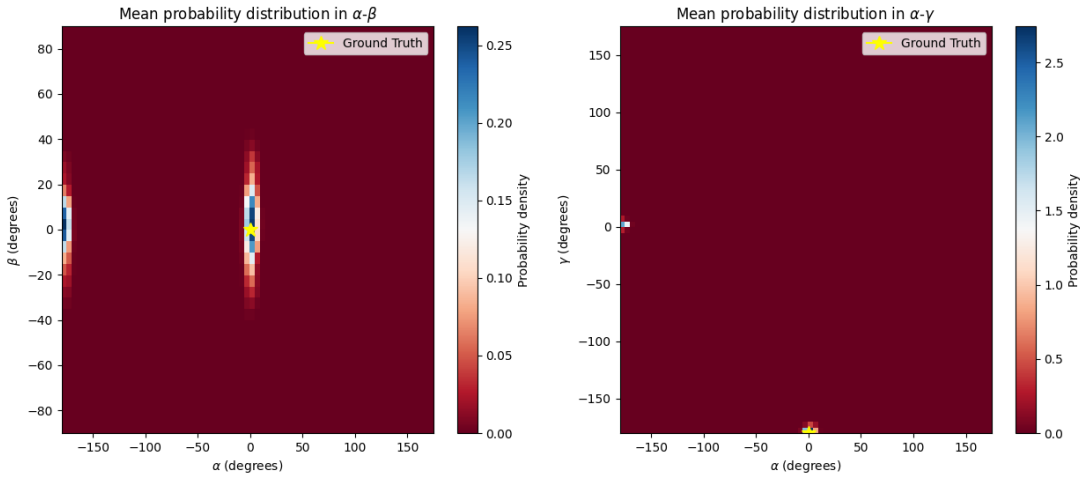
In this sub-section, we evaluate the contribution of each weight component to the results of our approach. To assess the results of each weight component we performed the filter under two different conditions. Performing the filter with a weight calculated as the YOLO-based likelihood (equation (15)) we obtained the results for the YOLO-based weighting. Instead, the autoencoder-base weighting results were obtained by performing the filter using as weight the autoencoder-based likelihood (equation (16)). The autoencoder-based weighting has a similar implementation to the baseline method [3].

First we performed the TransRot dataset, whose resust are shown in Table III. We observe that the autoencoder-based weighting is robust to handle a linear motion of the translation. Furthermore, this weighting component gets the lowest translation estimation mean error. Instead, the YOLO-based weight component gets the highest errors both on translation and rotation. The results of our approach are comparable but worst than the results of the autoencoder-based weight. We can conclude that when the UAV follows a translation movement compatible with the motion model, the autoencoder-based weight is robust enough to successfully estimate the pose.

The results are different when evaluating the weight components in a non-linear trajectory. Table IV shows the numerical results on Santos *et al* [5] dataset. We start by observing that the autoencoder-based weighting process has the worst



(a) Likelihood distribution obtained to the rotation represented in the left image of Figure 2.



(b) Likelihood distribution obtained to the rotation represented in the right image of Figure 2.

Fig. 10: Representation of the observation likelihood distribution for two different symmetrical rotations. The ground truth rotation is represented as a yellow star.

results on the translation component of the pose estimation. As the autoencoder-based weight does not have any translation observation update, it takes more time to adjust the filter to new velocities. In contrast, the YOLO-based weight shows a better translation estimation. This weight component allows the filter to rapidly adjust to new velocities, contributing to increasing the robustness to non-linear trajectories. The results in Table IV show that our approach has the lowest errors in Santos *et al* [5] dataset. Combining the two weight components we get robust translation and rotation estimations.

V. Conclusion

In this article, we presented a new weighting process combining detection and the rotation distribution. Our results show that by combining the two weight components, our method has higher robustness to non-linear trajectories. Furthermore, we observed that each weight component adds different contributions to the full system. The autoencoder-based likelihood weighting forces the filter to prioritize translations leveraging to a better rotation distribution, contributing also to the filter stability. Instead, the YOLO-based likelihood updates the filter with translation observations, increasing the robustness to the non-linear trajectories. Comparing our results with [5], we verified that our method surpasses their results. Particularly, our results show a great improvement in the UAV translation estimation. Furthermore, our approach successfully handles the UAV's rotation ambiguities as it holds various plausible hypotheses. This work is easily extendable to track multiple UAVs and to handle longstanding occlusions, applying for example a Boosted Particle Filter [24], which can be done in a future work.

Acknowledgments

This work was supported by FCT with the LARSyS-FCT Project UIDB/50009/2020 and project VOAMAIS (PTDC/EEI-AUT/31172/2017, 02/SAICT/2017/31172).

TABLE III: Numerical results on the TransRot dataset.

Pose component		5% Percentile	25% Percentile	Median	75% Percentile	95% Percentile	mean	Standard deviation
Translation (m)	Our approach	**0.25**	1.34	**2.98**	5.46	11.92	4.03	3.61
	YOLO-based weighting	0.30	1.38	3.72	7.01	14.56	5.12	4.92
	Autoencoder-based weighting	0.27	**1.26**	**2.98**	**5.33**	**8.97**	**3.58**	**2.80**
Rotation (degrees)	Our approach	**0.89**	**1.85**	**3.06**	**5.23**	18.25	5.81	10.70
	YOLO-based weighting	10.90	56.29	95.67	127.57	165.81	91.41	47.79
	Autoencoder-based weighting	0.96	2.03	3.39	5.63	**15.54**	**5.53**	**7.73**

TABLE IV: Numerical results on the Santos *et al* [5] dataset.

Pose component		5% Percentile	25% Percentile	Median	75% Percentile	95% Percentile	mean	Standard deviation
Translation (m)	Our approach	**0.26**	**1.10**	**2.55**	**5.13**	**10.83**	**3.62**	**3.35**
	YOLO-based weighting	0.34	1.36	3.10	7.71	15.56	5.11	5.04
	Autoencoder-based weighting	0.45	2.25	5.01	9.53	16.87	6.51	5.45
Rotation (degrees)	Our approach	**3.29**	**6.07**	**7.89**	**10.10**	**17.60**	**9.10**	**8.26**
	YOLO-based weighting	9.53	36.09	81.67	95.42	146.00	74.07	42.15
	Autoencoder-based weighting	3.90	6.73	8.59	11.04	30.62	10.65	10.53

REFERENCES

[1] Directorate-General for Natural Resources, Safety and Maritime Services (DGRM). Maritime Zones under Portuguese Sovereignty and / or Jurisdiction. URL: https://www.dgrm.mm.gov.pt/en/web/guest/am-ec-zonas-maritimas-sob-jurisdicao-ou-soberania-nacional. (accessed: 03/27/2021)

[2] K. W. Williams. A summary of unmanned aircraft accident/incident data: Human factors implications. Civil Aerospace Medical Institute, Federal Aviation Administration, 2004.

[3] X. Deng, A. Mousavian, Y. Xiang, F. Xia, T. Bretl and D. Fox. PoseRBPF: A Rao-Blackwellized Particle Filter for 6D Object Pose Tracking. Robotics: Science and Systems (RSS), 2019.

[4] A. Doucet, N. Freitas, K. Murphy, S. Russell. Rao-Blackwellised particle filtering for dynamic Bayesian networks. In Sequential Monte Carlo methods in practice. Springer, 2001.

[5] N. Santos, V. Lobo, A. Bernardino. Directional Statistics for 3D Model-Based UAV Tracking. In IEEE Access, 2020.

[6] J. Redmon and A. Farhadi. YOLOv3: An Incremental Improvement. Technical report, ArXiv, 2018.

[7] Q. Fu, Q. Quan and K. Cai. Robust pose estimation for multirotor UAVs using off-board monocular vision. IEEE Transactions on Industrial Electronics, 2017.

[8] R. Jin, J. Jiang, Y. Qi, D. Lin and T. Song. Drone detection and pose estimation using relational graph networks. Sensors, 2019.

[9] A. Rozantsev, S. N. Sinha, D. Dey and P. Fua. Flight dynamics-based recovery of a UAV trajectory using ground cameras. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019

[10] S. Hinterstoisser, V. Lepetit, S. Ilic, S. Holzer, G. Bradski, K. Konolige and N. Navab. Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes. Asian conference on computer vision, 2012.

[11] Z. Cao, Y. Sheikh and N. K. Banerjee. Real-time scalable 6DOF pose estimation for textureless objects. IEEE International conference on Robotics and Automation (ICRA), 2016.

[12] W. Kehl, F. Manhardt, F. Tombari, S. Ilic, and N. Navab. Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again. IEEE International Conference on Computer Vision (ICCV), 2017.

[13] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. Robotics: Science and Systems (RSS), 2018.

[14] G. Billings and M. Johnson-Roberson. SilhoNet: An RGB Method for 6D Object Pose Estimation. IEEE Robotics and Automation Letters, 2019.

[15] S. Peng, Y. Liu, Q. Huang, X. Zhou and H. Bao. PVNet: Pixel-wise Voting Network for 6DoF Pose Estimation. IEEE conference on computer vision and pattern recognition, 2019.

[16] H. Tjaden, U. Schwanecke and E. Schömer. Real-time monocular segmentation and pose tracking of multiple objects. European conference on computer vision, 2016.

[17] L. Zhong and L. Zhang. A robust monocular 3d object tracking method combining statistical and photometric constraints. International Journal of Computer Vision, 2019.

[18] L. Zhong, Y. Zhang, H. Zhao, A. Chang, W. Xiang, S. Zhang and L. Zhang. Seeing Through the Occluders: Robust Monocular 6-DOF Object Pose Tracking via Model-Guided Video Object Segmentation. IEEE Robotics and Automation Letters, 2020.

[19] F. Manhardt, W. Kehl, N. Navab and F. Tombari. Deep model-based 6d pose refinement in rgb. Proceedings of the European Conference on Computer Vision, 2018.

[20] S. Thrun, W. Burgard, and D. Fox. Probabilistic Robotics. MIT Press, 2005.

[21] F. L. Markley, Y. Cheng, J. L. Crassidis, and Y. Oshman. Averaging quaternions. Journal of Guidance, Control, and Dynamics, 2007.

[22] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. International Conference on Learning Representations (ICLR), 2015.

[23] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. International Conference on Artificial Intelligence and Statistics, 2010.

[24] K. Okuma, A. Taleghani, N. De Freitas, J. J. Little, and D. G. Lowe. A boosted particle filter: Multitarget detection and tracking. European conference on computer vision, 2004.

[25] N. P. Santos, V. Lobo and A. Bernardino. AUTOLAND project: Fixed-wing UAV Landing on a FPB using Computer Vision. [Presentation] URL: https://www.researchgate.net/publication/338778808_AUTOLAND_project_Fixed-wing_UAV_Landing_on_a_FPB_using_Computer_Vision. (accessed: 07/26/2021)