Real-Time 3D Tracking of Simple Objects with an RGB Camera

Student Lino Pereira[1]
lino.cp95@gmail.com
Student Bernardo Ferreira[1]
bernardoppferreira@gmail.com
Prof Alexandre Bernardino[2]
http://www.isr.tecnico.ulisboa.pt/~alex

[1] Instituto Superior Técnico, 1049-001 Lisboa, Portugal
[2] Institute for Systems and Robotics, Instituto Superior Técnico, 1049-001 Lisboa, Portugal

Abstract

This work, intends to improve a monocular region-based tracking algorithm using an RGB camera. The algorithm to be improved, derives from a particle filter where each particle represents a hypothesis of the state of the object in 3D. However, the literature mentions that the particle filter (PF) uses a very limited importance distribution to propagate the particles, which easily leads the filter to degenerate and loose track of the object. Given the limitation of the PF, an unscented particle filter (UPF) is proposed. This one obtains an approximation to the optimal importance distribution, by adding a current observation of the state.

In order to compare the proposed algorithm with the previous one, both are implemented and several real and simulated experiments with a simple object are performed. From the results obtained, is shown that the filters are successful, with the UPF being more robust.

1 Introduction

The most known methods to track an object using an RGB camera, are the methods based on 3D reconstruction and the ones based on test hypothesis. The first methods, start by using the visual information of the 2D image to reconstruct the pose of the object in 3D and the other ones, consists in generating numerous hypothesis about what could be the exact state of the object in 3D, and test each hypothesis from the 2D image information. The advantages of 3D reconstruction is that it's fast and easy to localize the object of interest, however, they are easily affected by noise in the image. On the other hand, the last methods are more precise because the image's noise is not taken into account, yet, they are very slow and poor in localizing the object. The main intent to use the UPF is to combine both methods to get the advantages of both. Thus, by defining the particles as 3D object's state hypothesis and introducing a current measurement of the object's state through a 3D reconstruction process, an hybrid algorithm is formulated.

In Bayes perspective and under the Markov assumption, the problem is to recursively estimate the posterior distribution of the current state $\mathbf{x}_t$ conditioned on all available observations $\mathbf{z}_{1:t} = \{\mathbf{z}_1, ..., \mathbf{z}_t\}$. One just needs to define some initial prior $p(\mathbf{x}_0)$, state transition $p(\mathbf{x}_t|\mathbf{x}_{t-1})$, and observation $p(\mathbf{z}_t|\mathbf{x}_t)$ probabilities, in mathematical terms [5]:

$$p(\mathbf{x}_t|\mathbf{z}_{1:t}) \propto p(\mathbf{z}_t|\mathbf{x}_t) \int p(\mathbf{x}_t|\mathbf{x}_{t-1}) p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1}) d\mathbf{x}_{t-1}. \quad (1)$$

However, the equation (1) is intractable, and for this reason many kind of numerical approximations, like the methods based on particles, have been developed. They represent the posterior distribution as N weighted set of Monte Carlo samples $\left\{\mathbf{x}_t^{(i)}, w_t^{(i)}\right\}$, $i = 1, ..., N$, also known as particles, and by the law of the big numbers, the bigger the number of particles the lower is the variance of the approximation error [6]. Unfortunately, it's often impossible to sample directly from the posterior distribution, so a known and easy-to-sample distribution $q(\mathbf{x}_t^{(i)}|\mathbf{x}_{0:t-1}^{(i)}, \mathbf{z}_{1:t})$, called importance distribution is applied. By drawing samples from this distribution, a recursive estimate for the importance weights can be derived [6]:

$$w_t^{(i)} \propto \frac{p(\mathbf{z}_t|\mathbf{x}_t^{(i)}) p(\mathbf{x}_t^{(i)}|\mathbf{x}_{t-1}^{(i)})}{q(\mathbf{x}_t^{(i)}|\mathbf{x}_{0:t-1}^{(i)}, \mathbf{z}_{1:t})} w_{t-1}^{(i)}. \quad (2)$$

This type of methods exhibit a phenomenon called degeneration. This happens when some particles get all the weight and a lot of them get insignificant. To prevent this, a process of resampling is implemented to replicate the particles with high weights and discard the lower ones [6]. Doing this, brings more particles to regions of high likehood, which not only contributes to get better estimates, but also to avoid the particles from moving wrongly in the state space. One filter that derives from these type of methods, is the particle filter, that uses the simple state transition probability as the importance distribution. Yet, the literature mentions that in this type of methods, the most critical design issue is the choice of importance distribution. If the likelihood function is to narrow, or if it lies in one of the tails of the prior distribution, even the resampling process might not be enough to prevent degeneration [6]. In a Markov process, the optimal importance distribution in terms of minimizing the variance of the weights is given by:

$$q(\mathbf{x}_t|\mathbf{x}_{0:t-1}, \mathbf{z}_{1:t}) = p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{z}_t). \quad (3)$$

However, sampling from this distribution is non-trivial, because of the dependence on the atual observation $\mathbf{z}_t$, thus, with the intent of getting an approximation to this distribution, the unscented particle filter introduced by Van Der Merwe *et al.* [6] was developed. This one uses the unscented kalman filter (UKF), which introduces a current observation together with a Gaussian approximation of the state, as the importance distribution to propagate each particle [6]:

$$q(\mathbf{x}_t^{(i)}|\mathbf{x}_{0:t-1}^{(i)}, \mathbf{z}_{1:t}) = \mathcal{N}(\mu_t^{(i)}, \mathbf{P}_t^{(i)}), \; i = 1, ..., N. \quad (4)$$

2 Methodology

The algorithm to be improved in this work is the one developed by M. Taiana *et al.* [5], in which to track a homogeneous ball, the state is defined as the position and velocity of the ball in space $\mathbf{x}_t = \begin{bmatrix} x & y & z & \dot{x} & \dot{y} & \dot{z} \end{bmatrix}^T$. The algorithm is based on a particle filter, where each particle represents a 3D hypothesis of the ball's state, this allows one to overcome the inversion of the nonlinearity caused by the camera projection model and enables the use of realistic 3D motion models as the state transition probability [5]. On the observation model, each particle project a few tens of points onto the current image of a video frame from his state hypothesis, one set inside and the other outside the 3D object's silhouette. With the chromatic information of these points, a normalized color histogram for the inner region and another for the outer region, are constructed along with the normalized color histogram of the object's color model [5]. The likelihood of a particle is considered high, if the inner and model histograms are similar and at the same time, the inner and outer histograms are different. To express this mathematically, a metric $\mathcal{D}$ is constructed, based on the Bhattacharyya coefficient that quantifies the similarity between histograms. At last, the observation probability of each particle is modeled by a Laplacian distribution over the metric $\mathcal{D}$, where ε was set to $\varepsilon = 1/30$ [5]:

$$p(\mathbf{z}_t|\mathbf{x}_t^{(i)}) \propto e^{-\frac{\mathcal{D}}{\varepsilon}}. \quad (5)$$

2.1 Proposed algorithm

On the previous algorithm, a motion model is applied to predict the next state of the particles, for the UPF, it's the unscented kalman filter that is used. This filter returns a prediction considering a motion model and a current observation, where the observation is a measure of the 3D position of the ball. The measurement process, consists in a method to estimate the current 3D position of the ball from an image. In order to accomplish this, a few steps must take place. The method first starts with color segmentation to identify the whereabouts of the ball. The ball is identified in the image through the highest pixel probability, corresponding to the reference histograms created based on the ball's color model, and the image is then binarized with the use of Otsu threshold [3] to distinguish the ball from the background. Next, with the use of morphological operators, the possible noise that survived the threshold, is removed and the edges of the ball smoothed. The contour is extracted with the Moore Neighborhood [4] tracing algorithm and these points are used to extract an ellipse with the RANSAC [2] algorithm. With the best fitted ellipse, the 3D position is then obtained through monocular reconstruction [1] that uses the prior information of the ball radius and camera parameters to estimate a position from the ellipse fitted to the blob.

3 Results & Discussion

In order to access and compare the performance of the PF and UPF, several tests over simulated trajectories and real experiments were made. Results for a simulated circular trajectory and for a real free-fall trajectory are shown in this section. For both filters, there are adjustments parameters that affects the performance. The principal and only parameters tested are: the number of particles (the higher the number the better are the estimates), the distance between the inner and outer points (that controls the measurement error) and the process model noise (that regulates the dispersion of the particles in the state space). For all the plots, the tests were made using $N = 1024$ particles, where the red lines represent closer inner and outer points, the green corresponds to points a bit more distant than the red ones, and the blue even more distant. The solid, dashed and dotted lines, represent three different process noise configurations. For all tests, the motion model used corresponds to a constant velocity model. To analyze the influence of the number of particles, the root mean square error (RMSE) was used and to examine the influence of the silhouette distances and process noises, precision plots were created. This plots instead of the RMSE, can catch if a filter looses track of an object, and for filters like these, this scenario often happens. Precision plots express the percentage of estimates that possess an error below a given error threshold, as the error threshold increases. The considered error threshold is the relative error δ. Therefore, the following equation is used to compute the percentage of the estimates F, that possess an error below an arbitrary relative error δ:

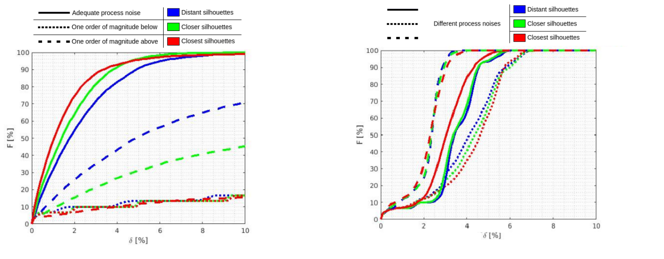
$$F = \frac{100}{N}\sum_{i=1}^{N} H\left(\delta - \frac{||\mathbf{x}_i - \hat{\mathbf{x}}_i||}{||\mathbf{x}_i||}\right) \quad [\%] \tag{6}$$

where H is the Heaviside function, N represents the number of estimates that belong to the experience, and $\mathbf{x}_i$ and $\hat{\mathbf{x}}_i$ corresponds respectively, to the exact state and the state estimate i. Both filters deal with random variables, thus, making tests with the same parameters originates different results and for this motive each test is repeated 100 times.

N	128	256	1024
PF	6.32×10^8	392.35	24.13
UPF	27.05	26.04	23.98

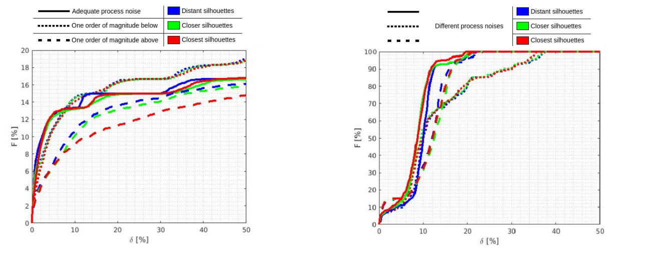
Table 1: RMSE error in *mm* using different number of particles.

In the table 1 are exposed results of RMSE of the position estimations for a given experience, varying only the number of particles. One can verify that the results coincides with the literature, once as the number of particle increases, better are the estimates, but the lower is the computational efficiency. Comparing the real results against the simulated ones (figure 1 and figure 2), it is quite visible, that for the real experiences, the relative error is bigger, at least the double, which make sense, because the simulator does not take into account the real phenomena of the world. Other aspect is the high sensitivity that the PF exhibits for different process noises and different distances between the inner and outer points (see figure 1(a)). For this filter, the process noise is directly related to the acceleration of the object and for one order of magnitude below or above the process noise used by the solid lines, the filter looses track of the ball and degenerates, which leads to wrong estimates. For the dotted lines the problem is the low scattering of the particles, and so, the filter cannot keep up with the object's movement. For the dashed lines, the particles get scattered too much and deviate from the ball which degenerates the filter. For the UPF the estimates of the position are all adequate for different process models. The real trajectory is a free-fall in which the ball collides with the ground multiple times, that makes the ball to rapidly change it's movement. That's why for the PF, the obtained results are very poor (see figure 2(a)). After an impact, the particles easily loose track of the ball and hardly recover to regions of high likelihood. On the other hand, one can see the real advantage of the UPF. If the particles loose track of the ball (mainly after an impact), the current observation acquired from a 3D reconstruction based method that easily localize the ball, will pull the particles to regions of high likelihood. Despite using such a limited motion model for this trajectory, the UPF obtains satisfactory results (see 2(b)).



(a) PF. (b) UPF.

Figure 1: Position estimates for the simulated circular trajectory.



(a) PF. (b) UPF.

Figure 2: Position estimates for the real free-fall trajectory.

4 Conclusions

The results obtained in this work, allows one to conclude that the implemented filters function with success, if the filters initial parameters are adjusted accordingly to the object's trajectory. For high uncertainty trajectories like a free-fall, the PF easily degenerates, contrarily, the UPF was successfully in all tests for any trajectory, which allows one to conclude that it's way more robust against all the three tested parameters. As future work, different observation models can be developed in order to make the algorithms usable for more complex objects.

Acknowledgements

This work was supported by:
FCT with the LARSyS - FCT Project UIDB/50009/2020.

References

[1] N. Greggio and J. Gaspar *et al.* Monocular vs binocular 3d real-time ball tracking from 2d ellipses. In *ICINCO 2011 - Proceedings of the 8th International Conference on Informatics in Control, Automation and Robotics*, volume 2, June 2011.

[2] K. Kanatani, Y. Sugaya, and Y. Kanazawa. Ellipse Fitting. In *Guide to 3D Vision Computation. Advances in Computer Vision and Pattern Recognition*, pages 11–32. Springer, Cham, 2016.

[3] N. Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1):62–66, 1979.

[4] P. Reddy and V. Amarnadh *et al.* Evaluation of Stopping Criterion in Contour Tracing Algorithms. *International Journal of Computer Science and Information Technologies*, 3(3):3888–3894, 2012.

[5] Matteo Taiana and Joao Santos *et al.* Tracking objects with generic calibrated sensors: An algorithm based on color and 3d shape features. *Robotics and Autonomous Systems*, 58(6):784 – 795, 2010.

[6] R. Van Der Merwe and A. Doucet *et al.* The unscented particle filter. In *Proceedings of the 13th International Conference on Neural Information Processing Systems*, page 563–569, Cambridge, MA, USA, 2000. MIT Press.