MULTI-CAMERA 3-D TRACKING USING PARTICLE FILTER

Pablo Barrera, José M. Cañas, Vicente Matellán, and Francisco Martín
Grupo de Robótica, Universidad Rey Juan Carlos
Móstoles, Madrid (Spain)
{barrera,jmplaza,vmo,fmartin}@gsyc.escet.urjc.es

Abstract

Determining the 3D location of a moving object, and tracking it from a sequence of different camera images is a classical but still challenging problem. In our approach neither explicit triangulation, nor precise motion model are used; only the colour of the object to be tracked is required. We use a particle filter, where the observation model we have developed avoids the colour filtering of the entire image. Preliminary experiments are presented and lessons learned are commented. The approach easily scales to several cameras and new sensor cues.

1. Introduction

Cameras are cheap and ubiquitous sensors. Images provide much data about the environment, but usually it takes lot of computing power to extract relevant information from them. One of the basic pieces of information they may provide is the 3D location of an object or person that is moving around the camera environment. Many commercial applications may take benefit from a robust object tracking. For instance, in security applications, unsupervised cameras may autonomously track moving persons and trigger an alarm if the person approached to any protected location.

The object tracking techniques are also used in robotics to cope with the self-localization problem. If the mobile robot tracked the relative 3D positions of some surrounding objects, and their absolute locations are known, then it could infer its own position in such absolute frame of reference. Such objects may not be dynamic, but the robot's motion causes a relative movement, which demands tracking (Davison's work [Davison04] is a good example for this). Actually, object tracking and localization share much mathematical background of dynamic state estimation, like Kalman filters, grid based methods and MonteCarlo sampling methods [Fox99, Arulampalam02].

The approach to 3D tracking we present in this paper uses a particle filter based on the CONDENSATION algorithm [Isard98], but in a different scenario from the contour tracking inside an image. Our algorithm uses only colour for 3D tracking from 2D observations (the images). It requires calibrated cameras and uses projective geometry to project particles in all the images. The colour of such projection and its neighbours provides feedback about the closeness of the particles to the real 3D location of the coloured object. A gaussian random noise is used as the motion model of the particles. Such model allows the particle population to follow any object movement. A similar approach [Perez04] has been recently followed to track objects inside images based on movement, colour and speech cues.

The remainder of the paper is organized as follows: next section explains our particle filter, detailing the observation and motion models used. Third section introduces the experimental setup and some tests of the system. Finally some conclusions and future lines are sketched out.

2. Colour based particle filter for 3-D tracking

The most popular techniques used in object tracking are Kalman filtering and MonteCarlo sampling approaches (apart from the simple triangulation). As an example of the first one, in [Davison03] a 2D tracking using Kalman filtering is proposed. They apply it to camera self-localization where some salient points are tracked along the monocular image stream.

Particle filters can be considered as an evolution of more traditional tracking techniques like Kalman filters. For instance in [Arulampalam02] Monte Carlo sampling techniques are used to explore and search in some state space. These sampling techniques improve the estimation performance and system robustness as multiple hypotheses are considered at the same time, as it has been pointed out in [Pérez04].

Our approach uses the CONDENSATION algorithm [Isard98] to estimate location of a coloured object. This is an iterative algorithm including three steps on each iteration: prediction, update, and resampling. CONDENSATION is a Bayesian recursive estimator that uses Sequential MonteCarlo Importance Sampling.

In short, it estimates the current multidimensional state $X(t)$, using a collection of sequential observations [$obs(t),obs(t\text{-}1),obs(t\text{-}2)...,obs(t_0)$]. The observations are related to the state through a probabilistic observation model $p(X(t)/obs(t))$.The state itself may be dynamic, and such dynamism is captured in a motion model $p(X(t)/X(t\text{-}1))$. The sequential nature of the algorithm provides iterative equations, and its sampling nature makes it to manage a set of N particles to represent the probability distribution over the set of plausible states, $p(X(t)/obs(t),obs(t\text{-}1)...)$. A more rigorous and broad description of probabilistic estimators can be found in [Mackay99, Arulampalam02].

Each particle $s_i(t)$ represents a state estimate and has a weight $w_i(t)$ associated, regarding the importance sampling. Global estimates can be made from the whole particle set, for instance choosing that of the higher weight (Maximum a Posteriori) or a weighted mean (Minimum Mean Square Error).

The *prediction step*, in each iteration of CONDENSATION, samples the motion model for every particle, obtaining a new $s_i(t)$, and so building a new particle set. In the *update step*, the weights of all particles are computed based on current images and following the observation model: $w_i(t)=p(s_i(t)/obs(t))$. Those particles that are likely given the current observation increase their weights. In the *resampling step*, a new set of particles is built sampling from the weighted distribution of current particles. The higher the weight, the more likely that particle appears in next set. Full details are provided at the original paper [Isard98].

In our approach the state to be estimated is the 3D location of the object $X=[x,y,z]$, so the particles have the shape of $s_i(t)=[x_i(t),y_i(t),z_i(t)]$, that is, they are 3D positions. The observations are just the colour images on M cameras and the motion model is just a simple one that randomly moves the particles through the 3D-space following a Gaussian distribution for each step.

2.1. Movement model

Using an approach similar to [Davison03], we have employed a weak motion modelling, in order to accommodate any real movement of the object. This provides robustness to the tracking algorithm as it avoids the need of a precise movement modelling to perform properly.

The motion model is a Gaussian distributed one, with the same typical deviation σ_m for x,y, and z-axis. It follows the equations (1), (2) and (3). There is no privileged motion direction, as the object may equally move in any of them. The size of σ_m has influence on the particle speed while walking through the state space.

$$x_i(t) = x_i(t-1) + N(0,\sigma_m) \quad (1)$$

$$y_i(t) = y_i(t-1) + N(0,\sigma_m) \quad (2)$$

$$z_i(t) = z_i(t-1) + N(0,\sigma_m) \quad (3)$$

2.2. Observation model

The update step gives the new weights of the particles according to the last sensor observation: $w_i(t)=p(s_i(t)|images(t))$. Our observation model is color based and works with any number M of cameras. It takes each camera separately, treating them as if they were independent observations and so multiplies all the partial conditioned probabilities:

$$w_i=\prod_{m=1}^{M} p_m(s_i(t)|img_m(t)) \quad (4)$$

Individual conditioned probabilities like $p_m(s_i(t)/img_m(t))$ are computed as follows. First, we project the 3D-particles into the corresponding image plane using a pinhole camera model. We assume cameras have no distortion.

- If such projection falls outside the image limits, then

$$outside: p_m(s_i(t)/img_m(t)) = 1/25 \quad (5)$$

- If the projection falls inside the image limits, its vicinity is explored to count the number k of pixels with a color similar to the target color. The vicinity is a 5x5 window around projected pixel. This can be seen in figure 1. The equation (6) assigns a probability proportional to k. To avoid probability locks with zeroes and to tolerate occlusions, k is set to 1 even if no pixel matches the target color description.

$$inside: p_m(s_i(t)/img_m(t)) = k/25 \quad (6)$$

The color is described in HSI space, which is more robust to changes in illumination than RGB. A target color is defined with two pairs, H_{min}, H_{max} and S_{min}, S_{max}. Pixels with very low or very high intensity are silently discarded and do not match any color description.

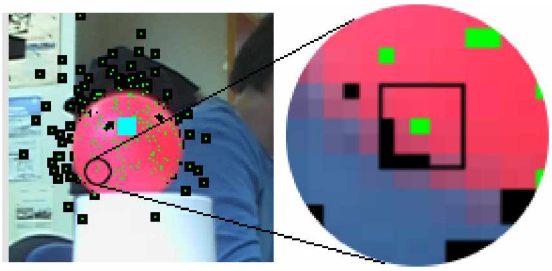

Figure 1: 5x5 vicinity window for observation model computation

The observation model in (4) clearly rewards those 3D locations that are compatible with several cameras simultaneously. In the case of two, the 3D locations compatible with one camera but which project badly in the other are score poorly, because $p_1(s_i(t)/img_1(t))$ or $p_2(s_i(t)/img_2(t))$ are set to a minimum, and that keeps the $w_i(t)$ at low values. This combined reward will lead the particles to the object position as only there the particles project correctly in both images.

Another advantage of this observation model is that it avoids the need to filter the color of the entire image. Depending on the number of particles this can be very convenient because it reduces the number of computations. In our experimental setup, for instance, filtering the whole image requires 320x240 pixel evaluations and the model requires Nx25 pixel evaluations. So for N<3072 it is worthwhile. In general, it would be worthwhile if N < *(nº image pixels / pixels in vicinity)*. In addition, the observation model does not require other time consuming task like segmentation in the images, the search for salient points or correspondences between images.

Our approach requires calibrated cameras, but no back-projection or triangulation is performed. Only the forward projections, from 3D particles into image planes, are used. Actually, there are no matching calculations between the stereo images, no correlation involved, and no explicit triangulation is carried out. As mentioned before, the observation model rewards those 3D locations that are color compatible in most of the images. This may include more space areas than the one where the object really is. This reflects the fact that particle filter can represent multiple simultaneous hypothesis about the state. New observations will eventually break the ambiguity and the population will converge to the real object position.

The developed algorithm is a true multi-image algorithm [Collins97]: there is no privileged camera, all images are treated equal and it may be used with an arbitrary number of cameras.

3. Experiments

Experiments with two cameras have been conducted in our laboratory. The observation model had been configured to track any pink object in a volume of 8 cubic meters, more volume than the previous experiments done in our laboratory [Barrera05]. The cameras are two webcams (M=2) providing 320x240 images, which have been calibrated using OpenCV library. Their external parameters like absolute position and orientation have been manually adjusted using a tape measure and projecting an absolute 3D grid into the images.

The particle filter has been tuned to 200 particles, and a typical deviation $s_m = 0.06$ (m). A typical filter iteration including all the prediction, update and resampling steps takes around 5 ms, on a Pentium IV, 2’7 GHz with HyperThreading, which is enough to real time performance.

Particles are randomly initialized in a cube of 2x2x2 meters. Then the system starts to search a pink object in a random walk over the space. When the object is found in one camera, a non-explicit epipolar search is performed until the object is located in both images. This process is shown in figure 3, displaying particle projection in both images at three different times.

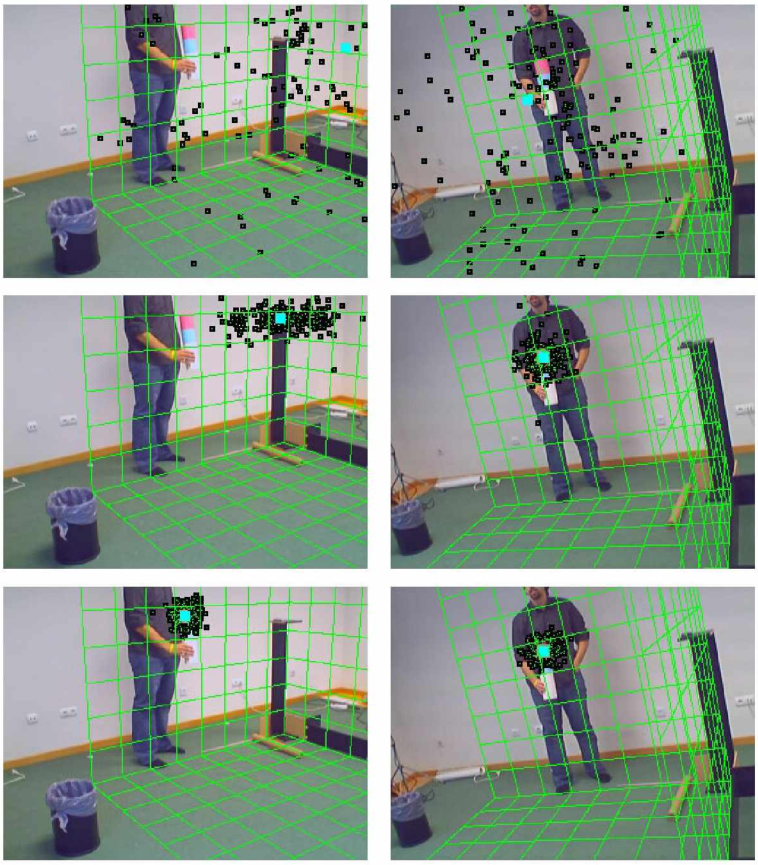

Figure 3: Particle projections in both cameras at three different times

The particles spread following the Gaussian motion model, searching the object over the space. As particles find it in one image they start to concentrate on that camera line of sight. Following that line particles start an epipolar-constrained search over the other camera until the real 3D positions is finally found. At that moment particles’ projections are coherent in both cameras. If the object does not move, the population remains stable around its real position. Once the population has converged, smooth movements of the object are successfully tracked in any direction. It can be noticed that convergence of the whole population speeds up as soon as some particles enter into the ball projection, due to the resampling step of the filter.

Manual measurements of the real object position were collected using a metric tape and compared to the system’s results. In all positions of the considered space, about eight cubic meters, the error was under 5cm. These results depend heavily on a proper calibration of extrinsic camera parameters. If these parameters are not correctly measured, there can be misalignments between the real space and the particle space, which lead to wrong estimation of object positions or to convergence only in one image at a time.

The particle filter maintains the estimation of the object positions and it makes predictions about its movement. This makes the system robust to temporal occlusions of the object in any camera or all of them. When the system cannot see the object in the images it grows the particle population to widen the search area until it finds the object again in one or both images.

Another results of experiments have been already explained in previous work, like the systematic pernicious trend of the particles to move far away from the cameras along their optical axis [Barrera05].

Some videos show how the system evolves, and they can be downloaded from the web[1].

4. Conclusions and Further Work

The work presented here summarizes the results on particle filter for object 3D tracking based on color information. The algorithm does not need any explicit triangulation or stereo matching at all, and it scales to an arbitrary number of cameras. The observation model used avoids the color filtering of the whole images and looks at the vicinity of the particle projections to estimate the particle's likelihood.

The results are promising, as convergence has been validated in real experiments in an eight cubic meter volume and the algorithm implementation exhibits real time performance. The real location of the object is a stable point for the particle cloud, the particles successfully track smooth movements of the object and the system is robust to short occlusions.

More experiments are necessary in order to validate the algorithm. Further improvements of the algorithm are coming. First, the use of more than two cameras simultaneously, in order to expand even more the volume inside which objects are successfully tracked. Second, we are also exploring some proposal distributions inside the filter, which hopefully would increase convergence speed of the cloud and its recovery capacity in case of losing the object. Third, we want to extend the system to track more than one object simultaneously.

References

[Isard 98] M. Isard and A. Blake, *CONDENSATION- conditional density propagation for visual tracking*, Int. Journal of Computer Vision, vol. 20, no. 1, pp. 5-28, 1998.

[Arulampalam02] S. Arulampalam, S. Maskell, N. Gordon and T. Clapp, *A tutorial on particle filters for on-line non-linear/non-gaussian bayesian tracking*, IEEE Transactions on Signal Processing, vol. 50, no. 2, pp. 174-188, 2002.

[Davison04] A.J. Davison, Y. González-Cid and N. Kita, *Real-time 3D SLAM with wide-angle vision*, 5th IFAC/EURON Symposium on Intelligent Autonomous Vehicles, July 2004.

[Davison03] A. J. Davison, *Real-time simultaneous localization and mapping with a single camera*, IEEE Int. Conf. on Computer Vision, ICCV-2003, pp. 1403-1410, Nice (France), October 2003.

[Pérez04] P. Pérez, J. Vermaak and A. Blake, *Data fusion for visual tracking with particles*, Proceedings of IEEE, vol. 92, no. 3, pp. 495-513, March 2004.

[Margaritis98] D. Margaritis and S. Thrun, *Learning to locate an object in 3D space from a sequence of images*. Proc. of Int. Conf. on Machine Learning, pp. 332-340, 1998.

[Louchet02] J. Louchet, M. Guyon, M.J. Lesot and A.M. Boumaza, *Dynamic flies: a new pattern recognition tool applied to stereo sequence processing*, Pattern recognition letters, vol. 23, pp. 335-345, 2002.

[Boumaza03] A.M. Boumaza and J. Louchet, *Mobile robot sensor fusion using flies* In Günter Raidl et.al. editors, Applications of Evolutionary Computing, EvoWorkshops 2003, Lecture Notes in Computer Science, vol. 2611, pp. 357-367, Springer, 2003.

[Collins97] R.T. Collins, *Multi-image focus of attention for rapid site model construction*, IEEE Int. Conf. on Computer Vision and Pattern Recognition, pp. 575-581, San Juan, Puerto Rico, 1997.

[Mackay99] D. Mackay, *Introduction to Monte Carlo methods*, In M. Jordan editor, Learning in graphical models, pp. 175-204, MIT Press, 1999.

[Fox99] D. Fox, W. Burgard, F. Dellaert and S. Thrun, *Monte Carlo localization: efficient position estimation for mobile robots*, In Proc. of 16th. AAAI Nat. Conf. on Artificial Intelligence, pp. 343-349, Orlando (USA), July 1999.

[Barrera05] Pablo Barrera, José M. Cañas and Vicente Matellán, *Visual 3D tracking using a color based particle filter*, In Proc. of 3rd. World Multi-Enformatika Conference pp. 200-203, Istanbul (Turkey), February 2005.

1 http://gsyc.escet.urjc.es/~jmplaza/research-visualtracking.html