ORIENTED PARTICLE SPRAY: PROBABILISTIC CONTOUR TRACING WITH DIRECTIONAL INFORMATION

François Pitié,[*] Anil C. Kokaram and Rozenn Dahyot
Electronic & Electrical Engineering
Trinity College Dublin
Ireland
email: fpitie@mee.tcd.ie, anil.kokaram@tcd.ie, dahyot@mee.tcd.ie

Abstract

Contour following is a standard activity in rotoscoping in the digital post production domain. An artist might need to *cut out* or edit an object separately from its background and it is left to the artist to manually create the cut out. Techniques for automatically tracing the edges of the object exist, but these operate with heavy manual intervention. The most recent technique called JetStream is a considerable advance on manual or semi-automatic tracing, but suffers from a lack of direction information in the image. This paper considers the incorporation of this information and so reworks the principle of density propagation for contour following. The approach is more robust than previous methods although inevitably needs user intervention to incorporate image semantics.
Keywords: *Particle Filter, Contour tracking, Rotoscoping, Bayesian Inference, Sequential Importance Resampling, Directional Filters, Steerable Filtering*

1 Introduction

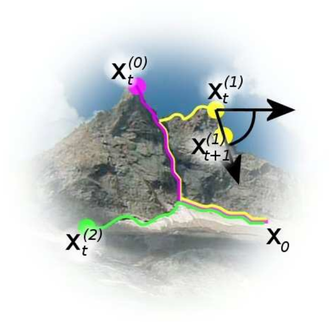
Manual or semi-automatic contour following is an important task in image editing. The tracing of object contours in general is also seen as an important task in early vision [3]. Cut-out tools that assist the user in following a contour, can be seen in Adobe Photoshop for instance. Automated or semi-automated contour following is complicated by the ambiguity of any contour in an image. Not only is it difficult to track exactly the position of a contour because of poor image contrast and noise, but also it is impossible to forsee the contour chosen by the user on the basis of semantics.

Recently, Perez, Blake and Gagnet [4] have proposed a robust technique—called *JetStream*—for contour following that handles this ambiguity by sampling from the posterior distribution for the contour location. It is based on the use of a Particle Filter and its operation can be understood as explained in the following section.

Probabilistic Tracing Approach using Particle Filters

The approach proposed in JetStream [4] to extract a contour can be understood by using an analogy with manual tracing. Starting from a point $\mathbf{x}_0$, the pencil draws a contour by following the edge of the picture. The current position of the pencil at time t is denoted $\mathbf{x}_t$. Tracing the contour can then be understood as *tracking* the pencil. The growing contour is represented by an ordered sequence $\mathbf{x}_{0:t} \equiv (\mathbf{x}_0 \dots \mathbf{x}_t)$.

[*]This work has been funded by HEA PRTLI TRIP and Enterprise Ireland Grant CASMS



Let θ_{t+1} be the angle formed by the segment $[\mathbf{x}_i ; \mathbf{x}_{i+1}]$ with the horizontal axis and let assume that the points are equally spaced by a step d. To simplify the problem, we assume that pencil speed is constant and therefore d is set to $d = 1$.

$$\mathbf{x}_{t+1} = \mathbf{x}_t + d \begin{bmatrix} \cos(\theta_{t+1}) \\ \sin(\theta_{t+1}) \end{bmatrix} \tag{1}$$

The idea of using Particle Filters for tracing is understood more easily with the help of the adjacent figure. While following a contour in the mountain picture, the pencil encounters bifurcations and edge junctions. To select the most likely path, the idea is to try all possible paths and to decide afterwards which one is the best. In our mountain picture example, growing contours $\mathbf{x}_{0:t}^{(0)}$ (in pink), $\mathbf{x}_{0:t}^{(1)}$ (in yellow) and $\mathbf{x}_{0:t}^{(2)}$ (in green) correspond to 3 different possible tracings all originating from the same starting point $\mathbf{x}_0$. The Particle Filter framework—described properly in the next section—proposes to grow simultaneously a number of possible contours—also called *particles*. The particles can take separate decisions when they reach an edge junction. The framework decides whether a particle should grow further, duplicate itself, or stop, depending on its performance.

JetStream, though an elegant solution to a combinatorially difficult problem, suffers from an inability to handle sudden changes in direction without the use of a switching process. In effect, upon encountering a corner, the idea is to propose unconstrained direction possibilities in the expectation that one of the proposed direction will regain a contour ‘lock’. This paper resolves the problem by designing a directional probability density function (pdf) that is better able to control the evolution of the contour. Because of the reliability of this pdf it is then possible to relieve the need for heavy control on contour smoothness. The particle filter framework is presented next and the new design explained as problems are highlighted.

2 Probabilistic Contour Tracking Framework

2.1 Standard Approach using Particle Filters

Recall that the ordered sequence $\mathbf{x}_{0:t} \equiv (\mathbf{x}_0 \dots \mathbf{x}_t)$ represents the 2D points of the curve being tracked. This chain is assumed to be a Markov Chain of order 2, ie. $p(\mathbf{x}|\mathbf{x}_{0:t}) = p(\mathbf{x}|\mathbf{x}_t, \mathbf{x}_{t-1})$. Given the observed image represented by a vector $\mathbf{y}$, a probabilistic approach to tracking proceeds by manipulating the posterior, $p(\mathbf{x}_{0:t+1}|\mathbf{y})$ to estimate the most probable next position $\mathbf{x}_{t+1}$. This distribution can be written in a recursive form:

$$p(\mathbf{x}_{0:t+1}|\mathbf{y}) = p(\mathbf{x}_{t+1}|\mathbf{y}, \mathbf{x}_{0:t})\, p(\mathbf{x}_{0:t}|\mathbf{y}) \tag{2}$$

This form admits a solution which manifests as the propagation of densities from point to point on each contour. Bayes rule combined with the Markovian hypothesis on the contour leads to the following expression for the posterior:

$$p(\mathbf{x}_{0:t+1}|\mathbf{y}) \propto \prod_{i=2}^{t+1} p(\mathbf{x}_i|\mathbf{x}_{i-1}, \mathbf{x}_{i-2})\, p(\mathbf{y}|\mathbf{x}_i, \mathbf{x}_{i-1}) \tag{3}$$

It is then possible to show that the following recursion arises:

$$p(\mathbf{x}_{0:t+1}|\mathbf{y}) = p(\mathbf{x}_{t+1}|\mathbf{x}_t, \mathbf{x}_{t-1})\, p(\mathbf{y}|\mathbf{x}_{t+1}, \mathbf{x}_t)\, p(\mathbf{x}_{0:t}|\mathbf{y}) \tag{4}$$

The term $p(\mathbf{x}_{t+1}|\mathbf{x}_t, \mathbf{x}_{t-1})$ corresponds to the *prior* on the contour and $p(\mathbf{y}|\mathbf{x}_{t+1}, \mathbf{x}_t)$ to the *data model*.

Although we might have an analytical expression for the prior and the data model, this expression presents usually no simple closed form. Sequential Monte Carlo methods (also called *particle filters*) provide however a flexible and easy way of propagating an approximation of this posterior distribution. In this framework the posteriors are approximated in a grid-based fashion by a finite set $(\mathbf{x}_{0:t}^{(m)})_{m=1...M}$ of M samples or *particles*:

$$p(\mathbf{x}_{0:t}|\mathbf{y}) \approx \sum_{m=1}^{M} w_t^{(m)} \delta(\mathbf{x}_{0:t} - \mathbf{x}_{0:t}^{(m)}) \tag{5}$$

where $\delta(.)$ denotes the Dirac delta measure which is 1 in 0 and zero otherwise; $w_t^{(m)}$ the importance weight attached to particle $\mathbf{x}_{0:t}^{(m)}$. Note that our particles correspond to contours $(\mathbf{x}_{0:t}^{(m)})$ and not to single 2D points. The posterior approximation can be propagated in time by the generic boostrap filter (or Sequencial Importance Resampling (SIR) Particle Filter) [1, 2] as proposed for instance in JetStream. At each time iteration, the weights are chosen using the principle of *importance sampling* [1, 2]. As we know, it can be difficult to draw directly samples from the posterior $p(\mathbf{x}_{0:t}|\mathbf{y})$. However, it is usually possible to find as a first step a proposal—called *importance density*—from which we can easily draw samples. In the bootstrap filter the proposal is simply the prior density $p(\mathbf{x}_{t+1}|\mathbf{x}_t^{(m)}, \mathbf{x}_{t-1}^{(m)})$ and the weights are therefore given by the likelihood [1, 2]:

$$w_{t+1}^{(m)} \propto \frac{p(\mathbf{x}_{t+1}^{(m)}, \mathbf{x}_t^{(m)}|\mathbf{y})}{p(\mathbf{x}_{t+1}^{(m)}|\mathbf{x}_t^{(m)}, \mathbf{x}_{t-1}^{(m)})} = p(\mathbf{y}|\mathbf{x}_{t+1}^{(m)}, \mathbf{x}_t^{(m)}) \tag{6}$$

To avoid that the weight distribution becomes more and more skewed which leads to the degeneracy of the particles, the bootstrap filter adds a *selection* step. In this crucial step the M growing contours are drawn from the normalised weight distribution. The idea is that 'good' contours will be statistically replicated whereas the 'bad' one will be deleted.

From these approximations of the posterior distribution $p(\mathbf{x}_{0:t}|\mathbf{y})$, an approximation of the Maximum A Posteriori can be derived by taking the 'best' contour.

2.2 Exact Importance Sampling

A good choice for the proposal is key to the success of the particle filter algorithm. In JetStream—as in many tracking algorithms—the importance distribution is however constrained by the smoothness of the particle's trajectory. For instance the trajectory of the contour cannot deviate by more than a few degrees. A special case is made when particles reach a corner: particles are allowed to take any direction. With such hypotheses the position of the next particle is strongly restricted and in our experience, at the price of missing frequently sharp turns in the contour as shown in figure 7. This problem arises due to the difficulty in designing a prior that will both play the role of a good proposal—able to restrict the search area—and that will give enough flexibility to model the dynamics of the contour.

As a key deviation from this classical approach, we propose to reconsider equation 3 and choose directly as the proposal

$$q(\mathbf{x}_{t+1}|\mathbf{y}, \mathbf{x}_{0:t}^{(m)}) = \frac{p(\mathbf{x}_{t+1}|\mathbf{x}_t^{(m)}, \mathbf{x}_{t-1}^{(m)})\ p(\mathbf{y}|\mathbf{x}_t^{(m)}, \mathbf{x}_{t+1})}{\int_{\mathbf{x}_{t+1}} p(\mathbf{x}_{t+1}|\mathbf{x}_t^{(m)}, \mathbf{x}_{t-1}^{(m)})\ p(\mathbf{y}|\mathbf{x}_t^{(m)}, \mathbf{x}_{t+1})\ d\mathbf{x}_{t+1}} \tag{7}$$

By doing so, we take the optimal proposal and we ensure a perfect sampling of the posterior, without any additional constraint on the prior function. The difficulty lies now in drawing the sample $\mathbf{x}_{t+1}^{(m)}$ directly from the proposal. Both prior $p(\mathbf{x}_{t+1}|\mathbf{x}_t^{(m)}, \mathbf{x}_{t-1}^{(m)})$ and likelihood $p(\mathbf{y}|\mathbf{x}_t^{(m)}, \mathbf{x}_{t+1})$ functions will be explicited in section 3 and section 4.

Figure 1: Outlines of the Oriented Particle Spray

1. **Initialisation.** $t = 0$, manually set $\mathbf{x}_0^{(m)} = \mathbf{x}_0$
2. **Importance Sampling Step**
   For each particle m, do:
   - **Prediction:**
   $$\mathbf{x}_{t+1}^{(m)} \sim p(\mathbf{x}_{t+1}|\mathbf{x}_t^{(m)}, \mathbf{x}_{t-1}^{(m)})p(\mathbf{y}|\mathbf{x}_t^{(m)}, \mathbf{x}_{t+1}) \tag{10}$$
   - **Weighting:**
   $$w_t^{(m)} = \int_{\mathbf{x}_{t+1}} p(\mathbf{x}_{t+1}|\mathbf{x}_t^{(m)}, \mathbf{x}_{t-1}^{(m)})p(\mathbf{y}|\mathbf{x}_t^{(m)}, \mathbf{x}_{t+1})d\mathbf{x}_{t+1} \tag{11}$$
3. **Selection Step.** Resample with replacement M contours from the set $\left(\mathbf{x}_{0:t+1}^{(m)}; m = 1, \dots, M\right)$ according to the normalised importance weights $w_t^{(m)} / \sum_m w_t^{(m)}$.

The weights are defined by

$$w_{t+1}^{(m)} \propto \frac{p(\mathbf{x}_{t+1}|\mathbf{y}, \mathbf{x}_{0:t}^{(m)})}{q(\mathbf{x}_{t+1}|\mathbf{y}, \mathbf{x}_{0:t}^{(m)})} \tag{8}$$

$$= \int_{\mathbf{x}_{t+1}} p(\mathbf{x}_{t+1}|\mathbf{x}_t^{(m)}, \mathbf{x}_{t-1}^{(m)})\, p(\mathbf{y}|\mathbf{x}_t^{(m)}, \mathbf{x}_{t+1})\, d\mathbf{x}_{t+1} \tag{9}$$

The final outline of our contour tracking algorithm is summarised in figure 1.

3 The Prior on the Contours

As the prior does not serve as a proposal, we can adopt a weak constraint on the dynamic of the contour. We only assume that a particle cannot return to a previous position. This problem—trivial in appearance—has to be handled carefully to avoid that the particles try to rediscover their exact reverse trajectory.

Using the trajectory angle θ, the prior can then be rewritten as

$$p(\mathbf{x}_{t+1}|\mathbf{x}_t^{(m)}, \mathbf{x}_{t-1}^{(m)}) = p(\theta_{t+1}|\theta_t^{(m)}) \tag{12}$$

We propose here a naive solution that disallows angles diametrically opposed to the previous direction angle taken by the particle.

$$p(\mathbf{x}_{t+1}|\mathbf{x}_t^{(m)}, \mathbf{x}_{t-1}^{(m)}) = \phi_b(\mathrm{dist}(\theta_{t+1}, \theta_t^{(m)})) \tag{13}$$

where ϕ_b is a kernel function based on the distance between angles as represented in figure 2.

4 Likelihood

Introducing the angle notation as previously, the likelihood can be reexpressed as

$$p(\mathbf{y}|\mathbf{x}_{t+1}, \mathbf{x}_t^{(m)}) = p(\mathbf{y}|\theta_{t+1}, \mathbf{x}_t^{(m)}) \tag{14}$$

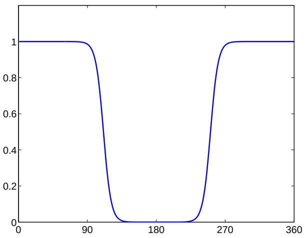


Figure 2: Values of the kernel for $\theta \to \phi_b(\mathrm{dist}(\theta, 0))$.

which stands for the probability that at pixel $\mathbf{x}_t^{(m)}$ an edge goes along the direction θ_{t+1}. The likelihood presented in JetStream relies mainly on the simple definition of the edge: the angle of the edge is defined by $\theta = \mathrm{atan2}(I_y, I_x)$[1] and its norm by $N = \sqrt{I_x^2 + I_y^2}$, where I_x and I_y are the derivatives of the picture I. This definition presents a strong drawback: it assumes that only one edge passes by the pixel of consideration. In consequence, this approach cannot cope with corners, or junctions. Even if JetStream attempts to handle this problem by using a Harris corner detector beforehand, Figure 7 shows that JetStream still tends to fail quite easily in its tracking. We propose therefore to fully integrate the orientation of the contours in our likelihood function. To do so, we make $p(\mathbf{y}|\theta_{t+1}, \mathbf{x}_t^{(m)})$ explicit by an approach similar to Steerable Filters [6, 5] and more specifically in [7].

Let us assume that the probability that at pixel $\mathbf{x}_t^{(m)}$, the direction θ_{t+1} corresponds to an edge is proportional to the absolute variation of the angular intensity, i.e.:

$$p(\mathbf{y}|\theta_{t+1}, \mathbf{x}_t^{(m)}) \propto \left| \frac{dI_\theta}{d\theta} \right| \tag{15}$$

where the intensity in direction $\theta \in [0; 2\pi]$ I_θ is equal to:

$$I_\theta = \int_{\rho>0} I(\rho, \theta) g(\rho) \, \mathrm{d}\rho \tag{16}$$

(ρ, θ) is a pixel coordinate location in polar coordinates, with origin at the current contour point. The integral is just the sum of pixels along the direction θ. $g(\rho)$ is a smoothing kernel (a gaussian for instance), which ensures that pixels closer to the origin are more important than those further away. Note that $\rho > 0$ since we wish to design a meaningful direction metric.

To interpolate I_θ to all values of θ we can take advantage of the periodicity of I_θ (since the function would repeat every 360^{deg}) and so consider its Fourier series:

$$I_\theta = \sum_{n=0}^{n=N} \mathrm{H}_n e^{jn\theta} \tag{17}$$

and respectively for its derivative:

$$p(\mathbf{y}|\theta_{t+1}, \mathbf{x}_t^{(m)}) \propto \left| \frac{dI_\theta}{d\theta} \right| = \left| \sum_{n=0}^{n=N} n \; j \; \mathrm{H}_n e^{jn\theta} \right| \tag{18}$$

The Fourier coefficients can be computed with:

$$\mathrm{H}_n = \int_{\phi,\rho} I(\rho, \phi) w_n(\rho, \phi) \; \rho \mathrm{d}\phi \mathrm{d}\rho \tag{19}$$

[1] $\mathrm{atan2}$ is $\tan^{-1}$ with unwrapped angles.

where

$$w_n(\rho,\phi) = \frac{1}{\rho} g(\rho) e^{jn\phi} \tag{20}$$

The continuous values of $I(\rho,\theta)$ are obtained by interpolation from the image grid. This can be classically obtained by convolving the sampled picture $\mathrm{I}(x,y)$ with an interpolation kernel k. To simplify notations we will consider cartesian coordinates:

$$\begin{cases} (u,v) & \equiv & (\rho,\phi) = (\sqrt{u^2+v^2}, \mathrm{atan2}(v,u)) \\ (x,y) & \equiv & (r,\psi) = (\sqrt{x^2+y^2}, \mathrm{atan2}(y,x)) \end{cases} \tag{21}$$

$$\begin{aligned} \mathrm{H}_n &= \int_{u,v} (\mathrm{I} * k)(u,v) w_n(u,v)\, \mathrm{d}u\mathrm{d}v & (22) \\ &= \int_{u,v} \left(\sum_{x,y} \mathrm{I}(x,y) k(u-x, v-y) \right) w_n(u,v)\, \mathrm{d}u\mathrm{d}v & (23) \\ &= \sum_{x,y} \mathrm{I}(x,y) \int_{u,v} k(u-x,v-y) w_n(u,v)\, \mathrm{d}u\mathrm{d}v & (24) \end{aligned}$$

By making explicit the interpolation kernel in this way, we are able to derive a complete framework for calculation of the direction information. Finally we have:

$$\begin{aligned} \mathrm{H}_n &= \sum_{x,y} \mathrm{I}(x,y) h_n(x,y) \\ h_n(x,y) &= \int_{u,v} k(u-x,v-y) w_n(u,v)\, \mathrm{d}u\mathrm{d}v \end{aligned}$$

So $\mathrm{H_n}$ can be computed by the use of a filter bank whose mask $h_n(x,y)$ can be computed offline. We still need to make explicit the kernels k and g. Here is a possible implementation:

$$\begin{aligned} g(\rho) &= \frac{1}{\sqrt{2\pi\sigma_g^2}} \exp\left(-\frac{\rho^2}{2\sigma_g^2}\right) \\ k(u-x,v-y) &= \frac{1}{2\pi\sigma_k^2} \exp\left(-\frac{\rho^2 + r^2 - 2r\rho\cos(\psi-\phi)}{2\sigma_k^2}\right) \end{aligned}$$

Figure 4, shows examples of 11-tap filters h_n.

Examples. Figure 5 shows an example of such a pdf. On the right the values of $\left|\frac{dI_\theta}{d\theta}\right|$ correspond to the pdf of the contour directions at the center of the picture on the left. This was obtained for $\sigma_g = 2.25$, $\sigma_k = 0.7$ at order $N = 10$. On the left side, the red lines correspond to the lobes of $\left|\frac{dI_\theta}{d\theta}\right|$.

5 Conclusion

Figure 7 shows some simulations of JetStream (on the left) and the Oriented Particle Spray (on the right). It is visible that JetStream tends to overshoot sharp angles of the contours whereas our method can follow them correctly, for a computational time equivalent to JetStream (the simulations where performed under matlab). This comparison has been carried out without user interaction that is an essential tool in a contour tracing application. The proposed improvements, in dealing better with sharp angles, should henceforth simplify and limit the user efforts.

A further development of this algorithm could be also to automatically extract *all* relevant contours of a picture by letting branches to grow separately after edge junctions.

Figure 3: Summary of the algorithm for computing the likelihood $p(\mathbf{y}|\theta_{t+1}, \mathbf{x}_t^{(m)})$

Offline computations:

$$h_n(x,y) \propto \int_{\rho,\phi} \exp\left(-\frac{\rho^2 + r^2 - 2r\rho\cos(\psi-\phi)}{2\sigma_k^2}\right) \exp\left(-\frac{\rho^2}{2\sigma_g^2}\right) \exp(jn\phi)\ \mathrm{d}\phi\mathrm{d}\rho \tag{25}$$

with the normalizing constant:

$$C = \frac{1}{2\pi\sigma_k^2\sqrt{2\pi\sigma_g^2}} \tag{26}$$

Online computations:

$$\mathrm{H}_n = \sum_{x,y} \mathrm{I}(x,y)h_n(x,y) \tag{27}$$

$$p(\mathbf{y}|\theta_{t+1}, \mathbf{x}_t^{(m)}) \propto \left|\frac{dI_\theta}{d\theta}\right| = \left|\sum_{n=0}^{n=N} jn\mathrm{H}_n e^{jn\theta}\right| \tag{28}$$

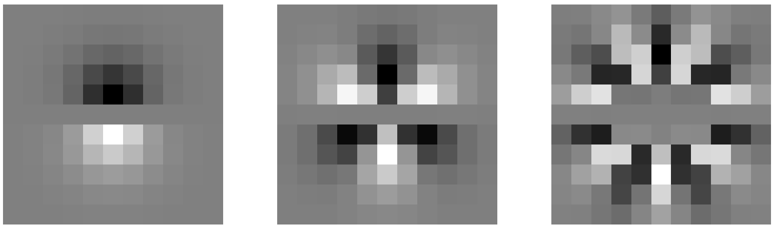


Figure 4: Examples of 11-tap filters $h_n(x,y)$ for $n=1$, $n=3$ and $n=7$.

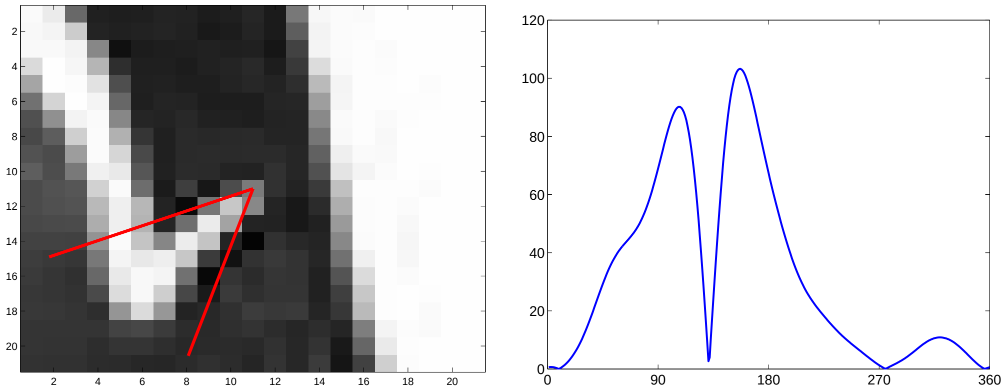


Figure 5: Example of an image (on the left) and the corresponding values of $\left|\frac{dI_\theta}{d\theta}\right|$ for θ in $[0°; 360°]$. On the left the red lines correspond to the directions of maximum variations (lobes of $\left|\frac{dI_\theta}{d\theta}\right|$).

References

[1] S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for on-line non-linear/non-gaussian bayesian tracking. 2002.

[2] Arnaud Doucet, Nando de Freitas, and Neil Gordon. *Sequential Monte Carlo Methods in Practice*. Springer, 2000.

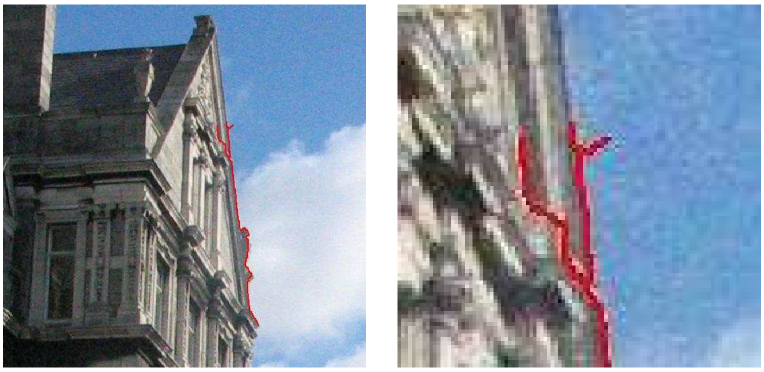


Figure 6: Example of the Oriented Particle Spray in action, with on the right a zoom on the multiple hypotheses tracking.

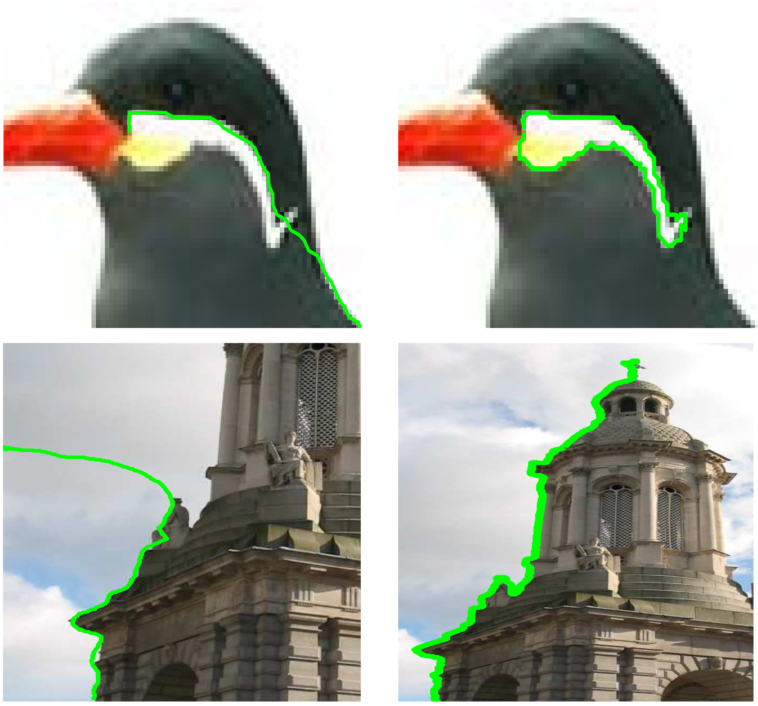


Figure 7: Contour tracings for JetStream on the left column and the Oriented Particle Spray on the right.

[3] David Marr. *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. W.H. Freeman, 1982.

[4] P. Pérez, A. Blake, and M. Gangnet. Jetstream: Probabilistic contour extraction with particles. *Proc. Int. Conf. on Computer Vision (ICCV)*, II(5):524–531, 2001.

[5] Pietro Perona. Deformable kernels for early vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(5):488–499, 1995.

[6] Eero P. Simoncelli and Hany Farid. Steerable wedge filters. In *ICCV*, pages 189–194, 1995.

[7] W. Yu, K. Daniilidis, and G. Sommer. Approximate orientation steerability based on angular gaussians, 1999.