Project Report: Find the difference! 01/09/2015

Jean Feydy
École Normale Supérieure
jean.feydy@ens.fr

Vincent Vidal
École Normale Supérieure
vincent.vidal@ens.fr

1. Introduction

With time, historical monuments may change and – slightly – evolve: thanks to old paintings or photos, we are able to appreciate it, and point at the differences. But what about computers? This is not a trivial problem at all. Indeed, two different pictures of the "same" building (e.g. a modern photo and an impressionist painting of Notre-Dame) may differ widely, as local textures or small architectural features are left subjects to the artist's good will.

In this work we shall see how SIFT Features can be used to detect occlusions, architectural deformations and drawing-errors, the main tool being the "SIFT Flow" algorithm proposed in [4].

We shall start in section 2 by giving a brief overview of the SIFT Flow algorithm. Then, we will discuss the possibility of using other features than SIFT to compute the optical flow, before describing in section 3 how the features flow can be used to detect occlusions. Eventually, in section 4, we will present a way to compute a quantitative evaluation of the distortion – "drawing errors" – for paintings and drawings.

2. SIFT Flow Algorithm

Basis Algorithm Being given two very different pictures of the same object (e.g. a monument), we would like to find a dense correspondence map between them, identifying regions in spite of variations in texture and appearance.

In order to reach this target, we first compute a dense feature field on each image, which gives a local description of every pixel's neighborhood – SIFT features are used in [4]. Using those features as a measure of similarity, we are then able to compute a deformation flow w, the "Feature flow": each pixel p in the first image is identified to the pixel $p + w_p$ of the second image. To compute a relevant w, we minimize the following energy function:

$$\begin{aligned} E(w) \quad = \quad & \sum_p \|\phi_1(p) - \phi_2(p + w_p)\|_1 \\ & + \gamma . \sum_p \|w_p\|_2^2 \\ & + \alpha . R(w,\, d) \end{aligned} \tag{1}$$

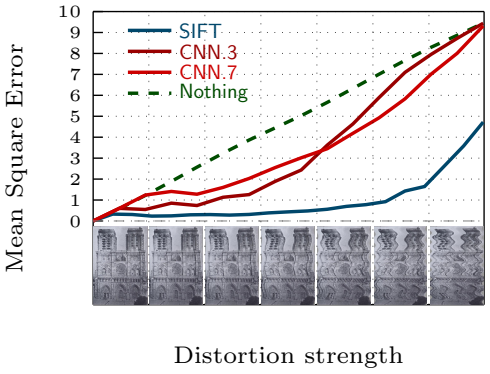


Figure 1. Error of the calculated field as a function of the distortion for the SIFT and CNN features. The error is the mean square error of the calculated deformation field. CNN3 corresponds to the 3rd layer of the CNN feature and CNN7 to the 7th layer.

with $\phi_i(p)$ the feature extracted from the image i at the pixel p, R a regularization function, and γ, α, d some real parameters.

The first term is there to enforce local similarity between the deformed source image and the targeted one.

The second term prevents w from being too large. It is related to the assumption that our two images are taken from roughly similar viewpoints: pixels shouldn't have to move too far away to find their perfect match – providing the artist is keeping the global structure of the picture intact.

The third term, the regularization term, is there to smooth the deformation field. We chose to keep the same function as [4] which correspond to a thresholded discrete gradient of the deformation field:

$$R(w) = \sum_{p,q \in \mathcal{N}} \min\big(|u_p - u_q|, d\big) + \min\big(|v_p - v_q|, d\big), \tag{2}$$

with $\mathcal{N}$ the set of neighboring pixels and u, v the components of w.

We used the code given in [4] that implements efficient belief propagation with some clever optimizations (see [2] for more details); please note that apart from this, all the code we used is ours.

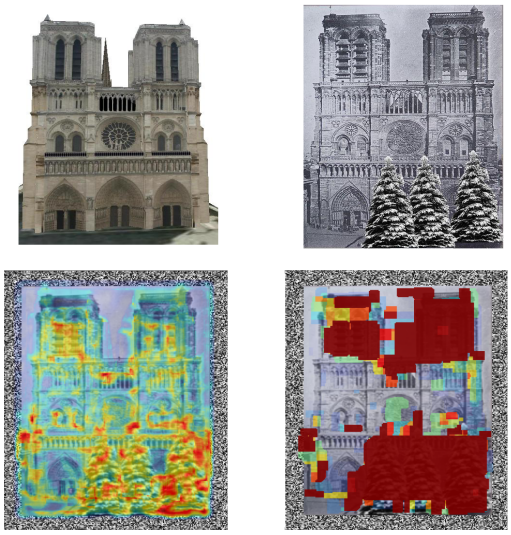


Figure 2. From top to bottom, left to right: Reference image, Target image, Method 1 (constrained) result and Method 2 (free-matching) result thresholded so that red corresponds to 10px displacement. Notice that the "occlusions" detected on the upper part of Notre-Dame correspond to actual architectural modifications.

Using CNN features A suitable path of improvement of SIFT flow is the introduction of CNN features. For that purpose, we use a "simple" neural network and choose one of the layers as the feature's map feeding the Flow algorithm.

In order to compare the efficiency of the different choices of features, we take an image, distort it with a known deformation v and measure the mean square error of the computed field. Figure 1 shows the error curves as a function of the distortion for the SIFT and CNN features.

The problem with choosing a layer as feature's map is that if we use a layer too deep, the size of the layer's grid will be too small and we will lose precision in the deformation field, which explains the bad performances in Figure 1 for small deformation, and if we use a shallower layer, the results are not that good. A good way to do so would be to extract a dense CNN features but a naive approach would take too much time; we would therefore need some deep optimization, which is the subject of [3].

3. Occlusions detection

The two strategies We now want to use SIFT flow to detect occlusions in a target image (e.g. an old painting) compared to a reference source image (e.g. a recent, clean photo). Our main idea is that occlusions should lead to outliers in the deformation field, that will be detected simply.

The energy function (eqn. 1) can be written as follows:

$$E(w) = E_{\text{Feat}} + \gamma . E_{\text{Norm}} + \alpha . E_{\text{Reg}}(d) . \qquad (3)$$

To detect occlusion, we developed two strategies. The first one is to constrain the deformation field to be smooth and small ($\gamma \sim 10^{-3}$), and detect occlusion by looking at the first term: "Feature's Energy". The second strategy consists in lightening the E_{Norm} term ($\gamma \sim 10^{-6}$), so that parts of the objects that are visible in the source image but occluded in the targeted one might be tempted to match other areas of the picture: in order to minimize the prevailing E_{Feat}, pixels from a brick wall occluded by a tree will tend to match brick regions elsewhere in the picture (which are very likely to be found) instead of mis-matching with the occluding foliage texture. We can then detect those movements by looking at the "Norm's Energy".

Experimentally, the second strategy produces results that are easier to use for detection purpose (see Figure 2). Indeed, as the first strategy roughly consists in taking the SIFT norm of the "difference" of the two images – the deformation flow correcting the small irregularities –, occlusions do not lead to plain high energy zones, but to noisy, cloudy areas. After all, the occluding mask may happen to have the same local aspect as the occluded area once every ten pixels.

Implementation First things first, we have to find parameters γ, α, d and P that lead to usable E_{Norm} maps – P is the patchsize of our SIFT descriptors. By hand (trying a large range of values for every parameter), we chose $P = 8$, $\alpha = 5.10^{-6}$, $\alpha = 2$, $d = 2$ – some small tuning through learning could have been done, but it wouldn't have lead to groundbreaking improvements as the most meaningful parameter in our case, P, is very constrained.

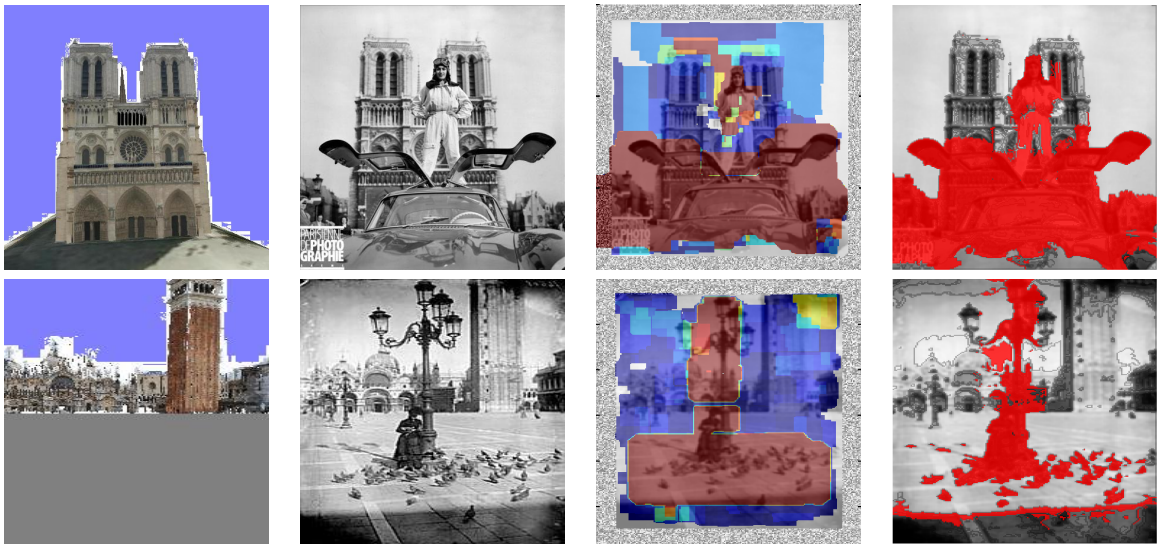
Without any treatment, sides and corners of our image are freer to move than its center, as a rectangle image can be "fold" without introducing any discontinuity. This behaviour makes little sense in the context of occlusion detection, and, instead of modifying the SIFT flow algorithm, we decided to solve this problem by using a framing trick: adding the same 20px-large border frame to the source and target image, we tend to fix the borders of the image, thus imposing the same continuity constraint on every pixel.

Eventually, not entirely satisfied by the rough shape of our occlusion regions (suplevel sets of E_{Norm}), we decided to combine it with an image segmentation algorithm based on Mean Shift [1]. Results can be seen in Figure 3, and would mainly benefit from a more efficient and entirely automatized segmentation technique.

4. Quantitative evaluation

We now want to get a quantitative evaluation of the deformation between the two images, which can be used to measure the accuracy of the representation.

A first way to do so could be to measure the norm of the deformation field. As we don't want to include a shifting part, we first remove the mean deformation to the deformation field and we don't want either to consider some occlud-



(a) Reference image (b) Occluded image (c) Norm Energy (d) Detected occlusions

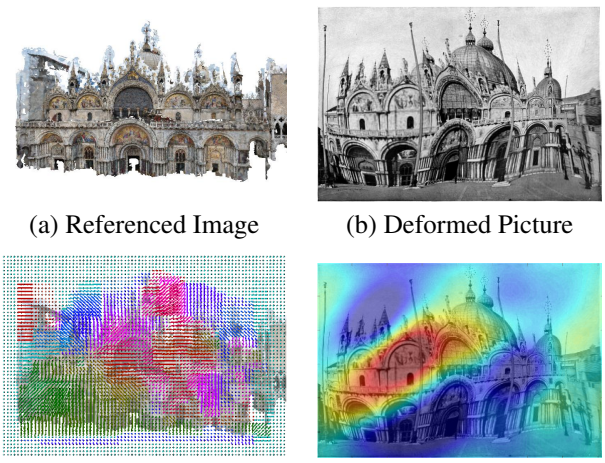
Figure 3. Occlusion's detection using the Norm Energy of SIFT Flow. (a) is the source image. (b) is the occluded target image. (c) is the representation of the norm energy (red threshold corresponds to a 10px displacement), with $P = 8$, $\alpha = 5.10^{-6}$, $\alpha = 2$, $d = 2$. (d) shows the detected occlusion, obtained by thresholding (c) and applying some segmentation method.

ing objects we need first to remove the occluded part of the deformation field.

We could also need a more local evaluation of the deformation: we want to be able to detect local contraction and extension of the deformation field. In order to get this information, we measure the gradient on a blurred field, to reduce the noise, and calculate a kind of divergence, as shown in Figure 4:

$$\mathrm{D}\,(w) = \frac{\partial u}{\partial x} + \frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} + \frac{\partial v}{\partial y} \quad \text{with} \quad w = (u,\, v)\,. \quad (4)$$

We notice that the blur variance parameter corresponds to the typical scale of the observed deformation.



(a) Referenced Image (b) Deformed Picture

(c) Deformation field (d) Deformation label

Figure 4. Painting deformation using SIFT Flow. (a) shows the referenced image. (b) shows the deformed picture from (a). (c) display the deformation field obtained with the SIFT Flow. (d) shows the divergence of the gradient of the deformation (red corresponds to dilatation, blue to shrinking).

5. Conclusion

We illustrated here the use of the feature flow algorithm to solve a non-trivial problem, the comparison of pictures of a similar scene with a huge variation in rendering style. Despite obtaining convincing result, we would like to mention the work that is still to be done, especially in the dynamic estimation of the scale and segmentation parameters.

References

[1] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(5):603–619, 2002. 2

[2] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient belief propagation for early vision. *Int. J. Comput. Vision*, 70(1):41–54, Oct. 2006. 1

[3] F. N. Iandola, M. W. Moskewicz, S. Karayev, R. B. Girshick, T. Darrell, and K. Keutzer. Densenet: Implementing efficient convnet descriptor pyramids. *CoRR*, abs/1404.1869, 2014. 2

[4] C. Liu, J. Yuen, A. Torralba, J. Sivic, and W. T. Freeman. Sift flow: Dense correspondence across different scenes. In *Proceedings of the 10th European Conference on Computer Vision: Part III*, ECCV '08, pages 28–42, Berlin, Heidelberg, 2008. Springer-Verlag. 1