In Defense of Gradient-Based Alignment on Densely Sampled Sparse Features

Hilton Bristow and Simon Lucey

Abstract In this chapter, we explore the surprising result that gradient-based continuous optimization methods perform well for the alignment of image/object models when using densely sampled sparse features (HOG, dense SIFT, *etc.*). Gradient-based approaches for image/object alignment have many desirable properties – inference is typically fast and exact, and diverse constraints can be imposed on the motion of points. However, the presumption that gradients predicted on sparse features would be poor estimators of the true descent direction has meant that gradient-based optimization is often overlooked in favour of graph-based optimization. We show that this intuition is only partly true: sparse features are indeed poor predictors of the error surface, but this has no impact on the actual alignment performance. In fact, for general object categories that exhibit large geometric and appearance variation, sparse features are integral to achieving any convergence whatsoever. How the descent directions are predicted becomes an important consideration for these descriptors. We explore a number of strategies for estimating gradients, and show that estimating gradients via regression in a manner that explicitly handles outliers improves alignment performance substantially. To illustrate the general applicability of gradient-based methods to the alignment of challenging object categories, we perform unsupervised ensemble alignment on a series of non-rigid animal classes from ImageNet.

Hilton Bristow
Queensland University of Technology, Australia. e-mail: hilton.bristow@gmail.com

Simon Lucey
The Robotics Institute, Carnegie Mellon University, USA. e-mail: slucey@cs.cmu.edu

1 Notation

Before we begin, a brief aside to discuss notation in the sequel. Regular face symbols (*i.e.* n, N) indicate scalars, with lowercase variants reserved for indexing and uppercase for ranges/dimensions; lowercase boldface symbols (*i.e.* $\mathbf{x}$) indicate vectors; uppercase boldface symbols (*i.e.* $\mathbf{J}$) indicate matrices, and uppercase calligraphic symbols (*i.e.* $\mathcal{I}$) indicate functions. We refer to images as functions rather than vectors or matrices to indicate that non-integer pixels can be addressed (by sub-pixel interpolation). This is necessary since the output coordinates of warp functions can be real valued. The notation $\mathcal{I} : \mathbb{R}^{D\times 2} \to \mathbb{R}^D$ indicates the sampling of D (sub-)pixels. To keep notation terse – and hopefully more readable – we often vectorize expressions. Therefore, in many instances, functions have vector-valued returns, though we endeavour to be explicit when this happens (as above).

2 Introduction

The problem of object or image alignment involves finding a set of parameters $\Delta\mathbf{x}$ that optimally align an input image $\mathcal{I}$ to an object or image model,

$$\Delta\mathbf{x}^* = \arg\min_{\Delta\mathbf{x}} \mathcal{D}\{\mathcal{I}(\mathbf{x}+\Delta\mathbf{x})\} + \mathcal{A}\{\Delta\mathbf{x}\}. \tag{1}$$

Under this umbrella definition of alignment, we can instantiate particular models of optical flow, pose estimation, facial landmark fitting, deformable parts modelling and unsupervised alignment commonly encountered in computer vision. $\mathcal{D} : \mathbb{R}^D \to \mathbb{R}$ is the continuous loss function which measures the degree of fit of the image observations to the model. $\mathcal{I} : \mathbb{R}^{D\times 2} \to \mathbb{R}^D$ is the image function which samples the (sub-)pixel values at the given locations, and we use the shorthand $\mathbf{x} = [\mathbf{x}_1^T, \dots, \mathbf{x}_D^T]^T$, $\mathcal{I}(\mathbf{x}) = [\mathcal{I}(\mathbf{x}_1+\mathbf{x}_1), \dots, \mathcal{I}(\mathbf{x}_D+\mathbf{x}_D)]^T$ where $\mathbf{x}_i = [x_i, y_i]^T$ is the ith $x-$ and $y-$ discrete coordinates sampled on a regular grid at integer pixel locations within the continuous image function. $\mathcal{A} : \mathbb{R}^{2D} \to \mathbb{R}$ is the regularization function that will penalize the likelihoods of each possible deformation vector $\Delta\mathbf{x}$. For example in optical flow, deformation vectors that are less smooth will attract a larger penalty. In the case of parametric warps (affine, similarity, homography, *etc.*) the regularization function acts as an indicator function which has zero cost if the deformation vector adheres to the desired parametric warp or infinite cost if it does not. In reality the alignment function $\mathcal{D}$ can be as complicated as a support vector machine [7], mutual information [8], or deep network [24], or as simple as the sum of squared distances (SSD) between an input image and a fixed template.

Since pixel intensities are known to be poor estimators of object/part similarity, it is common in alignment strategies to instead use a feature mapping

function,

$$\Delta\mathbf{x}^* = \arg\min_{\mathbf{x}} \mathcal{D}\{\Phi\{\mathcal{I}(\mathbf{x}+\Delta\mathbf{x})\}\} + \mathcal{A}\{\Delta\mathbf{x}\} \quad (2)$$

where $\Phi : \mathbb{R}^D \rightarrow \mathbb{R}^{DK}$ is a non-linear transformation from raw pixel intensities to densely sampled sparse features such as HOG [7] or SIFT [14]. K refers to the number of channels in the feature. In HOG and SIFT for example, these channels represent gradient orientation energies at each pixel location. In general one can attempt to solve either alignment objective (Eq. 1 and 2) in one of two ways: (i) graph-, or (ii) gradient- based search. The choice is largely problem specific, depending on the type of alignment and regularization function. We expand upon these considerations in the following subsections.

An especially important factor for gradient-based search strategies is the accuracy of the linearization matrix function of the representation (whether raw pixel intensities or densely sampled sparse features) with respect to the deformation vector. The linearization matrix function, or gradient as it is often referred to in computer vision, attempts to estimate an approximate linear relationship between the representation function and the deformation vector $\Delta\mathbf{x}$ over a restricted set of deformations. In this chapter we attempt to answer the question: *does the accuracy of this linearization reflect its utility in gradient search alignment?*

We argue that gradient search alignment strategies are often needlessly dismissed, as the linearization of $\Phi\{\mathcal{I}(\mathbf{x})\}$ of most natural images is poor in comparison to that obtained from $\mathcal{I}(\mathbf{x})$. We demonstrate empirically and with some theoretical characterization, that in spite of the poor linearization approximations of sparse features like SIFT and HOG, they actually enjoy superior gradient search alignment performance in comparison to raw pixel representations. We believe this result to be of significant interest to the computer vision community.

3 Search Strategies for Alignment

Graph-Based Search

If computation time was not an issue one would simply exhaustively search all finite deformation vectors $\Delta\mathbf{x}$ in order to find the global minima. This brute force strategy is tractable for coarse-to-fine sliding-window detectors such as Viola & Jones [22], but intractable for nearly all other deformations of interest within the field of computer vision. If the set of allowable displacements $\Delta\mathbf{x}$ is discretized and the function of parameters $\mathcal{A}$ constrained to obey a particular graphical structure (*e.g.* tree, star or chain), efficient graph optimization methods such as dynamic programming (i.e. belief propagation)

can be applied to solve for the globally optimal dformation in polynomial time. A prominent example can be found in deformable parts models whose ensemble of local part detectors are restricted to search discrete translations and scales, but with additional constraints placed on the spatial layout of the parts. If the dependencies between the parts can be represented as a tree, one can search all possible deformations (in order to find a global minima) in a computationally tractable manner via dynamic programming [11].

Although graphical models have proven popular in the alignment literature, they still face a number of problems. Inference in graphical models is difficult and inexact in all but the simplest models such as tree- or star-structured graphs. For example, in the application of graph-based search to optical flow – termed SIFT Flow [13] – the regularization on the $2D$ smoothness of the flow prevents the allowable warps from being factored into a tree structure. Instead, the authors employ an alternation strategy of enforcing the smoothness constraint in the $x-$ and then $y-$ directions (each of which independently can be represented as a tree structure) using dual-layer belief-propagation to find an approximate global solution. In many other cases, simplified tree- or star-structured models are unable to capture important dependencies between parts, so are not representative of the underlying structure or modes of deformation of the object being modelled [27]. The limited expressiveness of these simple models prevents many interesting constraints from being explored, which has led to the study of discrete but non-graphical models [17].

Gradient-Based Search

An alternative strategy for solving Eqn. 1 in polynomial time is through non-linear continuous optimization methods. This class of approaches linearize the system around the current estimate of the parameters, perform a constrained/projected gradient step then update the estimate, iterating this procedure until convergence. We refer to this strategy in the sequel as *gradient-based*.

Gradient-based search methods can be categorized as deterministic or stochastic. Deterministic gradient estimation can be computed in closed form and is computationally efficient. This requires the alignment function to be continuous and deterministically differentiable. Stochastic gradient estimation involves the sampling of a function with respect to its parametric input in order to estimate a first or second order relationship between the function and the input and can be computationally expensive (especially when one is trying to establish second order relationships). Both methods, when applied to object or image alignment, employ stochastic gradient estimation methods at some level. Deterministic methods estimate stochastic gradients on the representation, and then leverage closed form first and second order

derivative information of the alignment function. Stochastic methods, however, estimate stochastic gradients directly from the objective [25].

Deterministic gradient-based search methods have a long history in alignment literature [5, 10, 12, 26]. The most notable application of this concept is the classic Lucas & Kanade (LK) algorithm [15], which has been used primarily for image registration. The approach estimates gradients stochastically on the image representation, and then employs deterministic gradients of the objective of the SSD alignment function, resulting in an efficient quasi-Newton alignment algorithm. Many variations upon this idea now exist in computer vision literature [4, 5, 1] for applying deterministic gradient search to object registration.

A good example of stochastic gradient-based search for object/image alignment can be found in the constrained mean-shift algorithm for deformable part alignment (made popular for the task of facial landmark alignment [19]). In this approach, stochastic gradients around the alignment objective are estimated independently for each part detector, from which a descent direction is then found that adheres to the learned dependencies between those parts. The focus in this chapter, however, will be solely on deterministic gradient-based methods due to their clear computational advantages over stochastic methods.

4 Linearizing Pixels and Sparse Features

As stated earlier, our central focus in this chapter is to first investigate how well sparse features like HOG and SIFT linearize compared to pixel intensities. To do this we first need to review how one estimates the representation's gradient estimate $\nabla\mathcal{R}(\mathbf{x}) : \mathbb{R}^{2D} \rightarrow \mathbb{R}^{D\times 2D}$ when performing the linearization,

$$\mathcal{R}(\mathbf{x} + \boldsymbol{\Delta}\mathbf{x}) \approx \mathcal{R}(\mathbf{x}) + \nabla\mathcal{R}(\mathbf{x})\boldsymbol{\Delta}\mathbf{x} \tag{3}$$

where $\mathcal{R}(\mathbf{x}) \in \{\mathcal{I}(\mathbf{x}), \Phi\{\mathcal{I}(\mathbf{x})\}\}$ is a representation function that is agnostic to the choice of representation: raw pixel intensities $\mathcal{I}(\mathbf{x})$, or densely sampled sparse features $\Phi\{\mathcal{I}(\mathbf{x})\}$.

Gradient Estimation as Regression

One can view the problem of gradient estimation naively as solving the following regression problem,

$$\nabla\mathcal{R}(\mathbf{x}) = \arg\min_{\mathbf{J}} \sum_{\boldsymbol{\Delta}\mathbf{x}\in\mathbb{P}} \eta\{\mathcal{R}(\mathbf{x} + \boldsymbol{\Delta}\mathbf{x}) - \mathbf{J}\boldsymbol{\Delta}\mathbf{x}\} \tag{4}$$

where $\mathbb{P}$ is the set of deformations over which we want to establish an approximately linear relationship between the representation $\mathcal{R}(\mathbf{x} + \boldsymbol{\Delta}\mathbf{x})$ and the deformation vector $\boldsymbol{\Delta}\mathbf{x}$. η is the objective function used for performing the regression, for example $\eta\{.\} = ||.||_2^2$ would result in least-squares regression. This gradient estimation step can be made more efficient by considering each coordinate in $\mathbf{x} = [\mathbf{x}_1^T, \dots, \mathbf{x}_D^T]^T$ to be independent of each other. This results in a set of KD regression problems,

$$\nabla\mathcal{R}_i^k(\mathbf{x}_i) = \arg\min_{\mathbf{J}} \sum_{\boldsymbol{\delta}\in\mathbb{L}} \{\mathcal{R}_i^k(\mathbf{x}_i + \boldsymbol{\delta}) - \mathbf{J}\boldsymbol{\delta}\}, \;\; \forall\, i = 1:D, \;\; k = 1:K \tag{5}$$

where $\nabla\mathcal{R}_i^k(\mathbf{x}_i) : \mathbb{R}^2 \to \mathbb{R}^1$, $\mathbb{L}$ is the local translation deformation set for each pixel coordinate (normally a small window of say 3×3 or 5×5 discrete pixel coordinates), D is the number of pixel coordinates and K is the number of channels in the representation (e.g. for raw pixel intensities $K = 1$). We can then define $\nabla\mathcal{R}_i(\mathbf{x}_i) : \mathbb{R}^{2D} \to \mathbb{R}^{DK\times 2D}$ as,

$$\nabla\mathcal{R}(\mathbf{x}) = \begin{bmatrix} \nabla\mathcal{R}_1^1(\mathbf{x}_1) & & & \\ \vdots & & & \\ \nabla\mathcal{R}_1^K(\mathbf{x}_1) & & & \\ & \ddots & & \\ & & \nabla\mathcal{R}_D^1(\mathbf{x}_D) & \\ & & \vdots & \\ & & \nabla\mathcal{R}_D^K(\mathbf{x}_D) & \end{bmatrix}. \tag{6}$$

Of course, linear regression is not the only option for learning the gradient regressor. One could also consider using support vector regression (SVR) [9], which has better robustness to outliers. Intuitively, support vector regression predicts the gradient direction from a different weighted combination of pixels within a local region around the reference pixel. SVR has a clear performance advantage, with a commensurate increase in computation during training.

Gradients Estimation as Filtering

For a least-squares objective $\eta\{.\} = ||.||_2^2$ the solution to each gradient matrix function can be computed in closed form,

$$\nabla\mathcal{R}_i^k(\mathbf{x}_i) = \left(\sum_{\boldsymbol{\delta}\in\mathbb{L}} \boldsymbol{\delta}\boldsymbol{\delta}^T\right)^{-1} \left(\sum_{\boldsymbol{\delta}\in\mathbb{L}} \boldsymbol{\delta}[\mathcal{R}_i^k(\mathbf{x}_i) - \mathcal{R}_i^k(\mathbf{x}_i + \boldsymbol{\delta})]\right). \tag{7}$$

There are a number of interesting things to observe about this formulation. The first term in the solution is independent of the representation – it depends only on the local deformations sampled, and so can be inverted once rather

than for each $\mathcal{R}_i^k$. The second term is simply a sum of weighted differences between a displaced pixel, and the reference pixel, *i.e.*,

$$\begin{bmatrix} \sum_{\Delta x} \sum_{\Delta y} \Delta x (\mathcal{R}_i^k(x_i + \Delta x, y_i + \Delta y) - \mathcal{R}_i^k(x, y) \\ \sum_{\Delta x} \sum_{\Delta y} \Delta y (\mathcal{R}_i^k(x_i + \Delta x, y_i + \Delta y) - \mathcal{R}_i^k(x, y) \end{bmatrix} . \tag{8}$$

If $\boldsymbol{\delta} = [\Delta x, \Delta y]^T$ is sampled on a regular grid at integer pixel locations, Eqn. 8 can be cast as two filters – one each for horizontal weights Δx, and vertical weights Δy,

$$f_x = \begin{bmatrix} x_{-n} & \dots & x_n \\ \vdots & & \\ x_{-n} & \dots & x_n \end{bmatrix} \qquad f_y = \begin{bmatrix} y_{-n} & \dots & y_{-n} \\ \vdots & & \\ y_n & \dots & y_n \end{bmatrix} \tag{9}$$

Thus, an efficient realization of Eqn. 7 of the gradient at every pixel coordinate is,

$$\nabla \mathcal{R}_i^k(\mathbf{x}_i) = \left(\sum_{\boldsymbol{\delta} \in \mathbb{L}} \boldsymbol{\delta} \boldsymbol{\delta}^T \right)^{-1} \text{diag}\Big(\begin{bmatrix} f_x * \mathcal{R}_i^k(\mathbf{x}) \\ f_y * \mathcal{R}_i^k(\mathbf{x}) \end{bmatrix} \Big) \tag{10}$$

where $*$ is the $2D$ convolution operator. This is equivalent to blurring the image with a clipped quadratic and then taking the derivative. It is also possible to place weights on $\boldsymbol{\delta}$ stemming from $\mathbb{L}$ as a function of its distance from the origin. In the case of Gaussian weights this results in the classical approach to estimating image gradients by blurring the representation with a Gaussian and taking central differences. It is surprising that the two formulations make opposing assumptions on the importance of pixels, and as we show in our experiments section the clipped quadratic kernel induced by linear regression is better for alignment.

Pixels versus Sparse Features

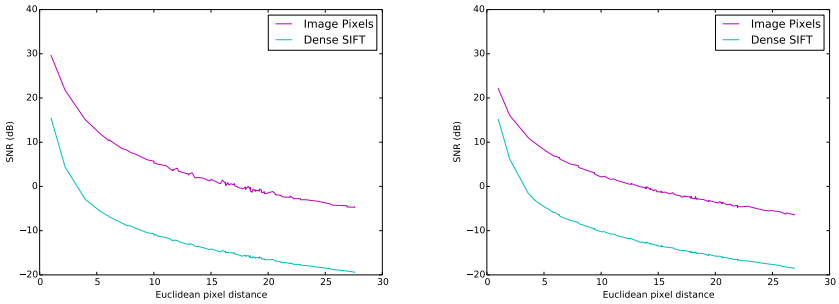
Considerable literature has been devoted to finding image features for general object classes that are discriminative of image semantics whilst being tolerant to local image contrast and geometric variation. The majority of existing feature transforms encode three components: (i) non-linear combinations of pixels in local support regions, (ii) multi-channel outputs, and (iii) sparsity. Prominent image features that exhibit these properties include HOG [7] and densely sampled SIFT descriptors [14]. We refer to this class of transforms as *densely sampled sparse features*.

Natural images are known to stem from a $\frac{1}{f}$ frequency spectrum [20]. This means that most of the energy in the image is concentrated in the lower frequencies – the image function is naturally smooth. Sparse multi-channel features follow no such statistics. In fact, they often exhibit highly non-linear

properties: small changes in the input can sometimes produce large changes in the output (*e.g.* gradient orientations close to a quantization boundary in HOG/SIFT can cause the output feature to jump channels, pixel differences close to zero in binary features can cause the output feature to swap signs), and other times produce no change in the output (*e.g.* orientations in the center of a bin, pixel differences far from zero).

To evaluate the generative capacity of different representations (*i.e.* how well the tangent approximation predicts the true image function at increasing displacements) we performed a simple experiment. We evaluated the signal-to-noise (SNR) ratio of the linearization function $\nabla\mathcal{R}(\mathbf{x})$ for increasing displacements across a number of images,

$$\text{SNR}(\mathbf{x}) = 10\log_{10}\left(\frac{||\mathcal{R}(\mathbf{x}+\boldsymbol{\Delta}\mathbf{x})||}{||\mathcal{R}(\mathbf{x})+\nabla\mathcal{R}(\mathbf{x})\boldsymbol{\Delta}\mathbf{x}-\mathcal{R}(\mathbf{x}+\boldsymbol{\Delta}\mathbf{x})||}\right)^2. \tag{11}$$



(a) (b)

Fig. 1 An experiment to illustrate the generative ability of pixel and densely sampled sparse features (in this case dense SIFT). We compute the linearization error $\mathcal{R}(\mathbf{x})+\nabla\mathcal{R}(\mathbf{x})\boldsymbol{\Delta}\mathbf{x}-\mathcal{R}(\mathbf{x}+\boldsymbol{\Delta}\mathbf{x})$ for a range of $\boldsymbol{\Delta}\mathbf{x}$ (x-axis), and look at the resulting signal-to-noise ratio (SNR) on the y-axis. The results are averaged over 10000 random trials across 100 images drawn from a set of (a) faces, and (b) animals. As expected, the generative accuracy of pixels is consistently higher than densely sampled sparse features, and better for face imagery than animal+background imagery (though the sparse representation is largely unchanged).

For simplicity, we restricted the deformation vectors $\boldsymbol{\Delta}\mathbf{x}$ to global translation. Fig. 1 illustrates the signal-to-noise ratio (SNR) versus Euclidean distance (*i.e.* $||\boldsymbol{\Delta}\mathbf{x}||_2$) for images of (a) faces, and (b) animals.

The tangent to the pixel image is a consistently better predictor of image appearance than the same applied to sparse features (in this case, dense SIFT). This confirms the intuition that pixel images are smoothly varying, whereas non-linear multi-channel features are not. The experiment of Fig. 1 indirectly suggests sparse features would not be appropriate for gradient-

based alignment search strategies. Unsurprisingly, graph-based optimization have become the strategy of choice for alignment when using sparse features, with some notable exceptions [16, 23, 25]. As a result, the wealth of research into continuous alignment methods [5, 12, 15, 18, 28] has largely been overlooked by the broader vision community.

5 Experiments

So far we have talked in generalities about gradient-based alignment methods, and the properties they enjoy. In this section, we instantiate a particular model of gradient-based alignment based on the Lucas & Kanade (LK) algorithm [15] in order to illustrate these properties in a number of synthesized and real-world settings.

We perform two tasks: (i) pairwise image alignment from a template image to a query image, and (ii) ensemble alignment where the alignment error of a group of images stemming from the same object class is minimized. In both tasks, existing gradient-based alignment approaches have typically only used pixel intensities, and as a result have only been evaluated on constrained domains such as faces, handwritten digits and building façades. Understandably, this has failed to capture the attention of the broader vision community working on challenging object classes with high intra-class variation.

We seek to show that gradient-based methods *can* be applied to object categories for which densely sampled sparse features are requisite to attaining meaningful similarities between images, and that a vanilla implementation of LK can go a long way to achieving interesting results on a number of challenging tasks.

The Lucas & Kanade Algorithm

Recollect our formulation of the alignment problem in Eqn. 1, this time using the representation function $\mathcal{R}$ that is agnostic to the choice of image function,

$$\Delta\mathbf{x}^* = \arg\min_{\Delta\mathbf{x}} \mathcal{D}\{\mathcal{R}(\mathbf{x}+\Delta\mathbf{x})\} + \mathcal{A}\{\Delta\mathbf{x}\}. \tag{12}$$

A common substitution within the LK algorithm is,

$$\mathbf{p}^* = \arg\min_{\mathbf{p}} ||\mathcal{R}(\mathbf{p}) - \mathcal{T}(\mathbf{0})||_2^2 \tag{13}$$

where $\mathbf{p}$ is a set of warp parameters that model the deformation vector $\Delta\mathbf{x}$ by proxy of a warp function,

$$\mathcal{R}(\mathbf{p}) = \begin{bmatrix} \mathcal{R}\{\mathcal{W}(\mathbf{x}_1;\mathbf{p})\} \\ \vdots \\ \mathcal{R}\{\mathcal{W}(\mathbf{x}_D;\mathbf{p})\} \end{bmatrix} \tag{14}$$

and $\mathcal{W}(\mathbf{x};\mathbf{p}) : \mathbb{R}^{2D} \rightarrow \mathbb{R}^P$. The warp function conditions the deformation vector on the warp parameters such that $\mathbf{x} + \Delta\mathbf{x} = \mathcal{W}(\mathbf{x};\mathbf{p})$. In most instances the dimensionality of $\mathbf{p} \in \mathbb{R}^P$ is substantially less than the canonical deformation vector $\Delta\mathbf{x} \in \mathbb{R}^{2D}$ (e.g. for a 2D affine warp $P = 6$). This is equivalent to setting $\mathcal{A}$ to be an indicator function, which has zero cost when the parameters fall within the feasible set of warps, and infinity otherwise. As in Eqn. 3, the LK algorithm takes successive first-order Taylor expansions about the current estimate of the warp parameters, and solves for the local update,

$$\boldsymbol{\Delta}\mathbf{p}^* = \arg\min_{\boldsymbol{\Delta}\mathbf{p}} ||\mathcal{R}(\mathbf{p}) + \nabla\mathcal{R}(\mathbf{p})\frac{\partial\mathcal{W}}{\partial\mathbf{p}}\boldsymbol{\Delta}\mathbf{p} - \mathcal{T}(\mathbf{0})||_2^2 \tag{15}$$

where $\nabla\mathcal{R}(\mathbf{p})$ is the gradient estimator, and $\frac{\partial\mathcal{W}}{\partial\mathbf{p}}$ is the Jacobian of the warp function which can be found deterministically or learned offline. Here we have presented the LK algorithm using the canonical L_2 loss function and the linearization function estimated from the input image representation $\mathcal{R}$ as opposed to the template $\mathcal{T}$. In reality there are a slew of possible variations on this classical LK form. [2] and [3] provide a thorough reference for choosing an appropriate update strategy and loss function. We present LK in this manner to avoid introducing unnecessary and distracting detail for the unfamiliar reader.[1] Regardless of these details, the choice of image representation and method of gradient calculation remain integral to the performance observed.

Pairwise Image Alignment

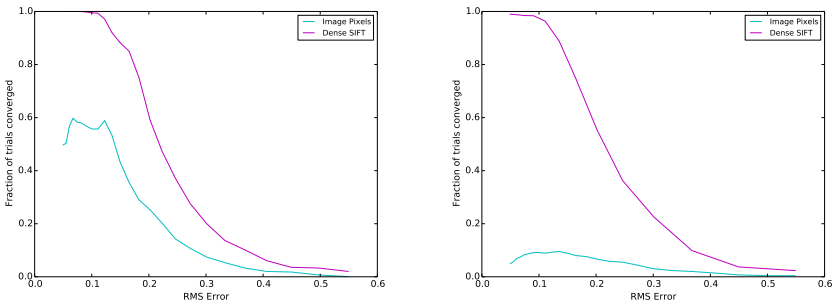
Earlier in Fig. 1 we performed a synthetic experiment showing the linearization error as a function of displacement for different image representations. Here we perform the sequel to that experiment, showing the frequency of convergence of the LK algorithm as a function of initial misalignment.

We initialize a bounding box template within an image, then perturb its location by a given RMS point error (measured from the vertices of the bounding box) and measure the frequency with which the perturbed patch converges back to the initialization after running LK. The results are show in

[1] In our experiments we actually estimate our linearization function from the template image $\mathcal{T}(\mathbf{0}) \rightarrow \nabla\mathcal{T}(\mathbf{0})$ using a technique commonly known within LK literature as the *inverse compositional* approach. This was done due to the substantial computational benefit enjoyed by the inverse compositional approach, since one can estimate $\mathcal{T}(\mathbf{0}) \rightarrow \nabla\mathcal{T}(\mathbf{0})$ once, as opposed to the classical approach of estimating $\mathcal{R}(\mathbf{p}) \rightarrow \nabla\mathcal{R}(\mathbf{p})$ at each iteration. See [2] and [3] for more details.

Fig. 2. We perform two variants of the experiment, (a) *intra*-image alignment, where the template and perturbation are sampled from the same image, and (b) *inter*-image alignment, where the perturbation is sampled from a different image of the same object class, with known ground-truth alignment. The task of inter-image alignment is markedly more difficult, since the objects within the template and the perturbation may have different non-rigid geometry, scene lighting and background clutter.

Even in the intra-image scenario, dense SIFT consistently converges more frequently than pixel intensities. In the inter-image scenario, the difference is even more pronounced. Fig. 3 shows a more comprehensive view of the inter-image scenario, with a comparison of the different gradient estimation techniques we have discussed. In general, there is a gradual degradation in performance from support vector regression (SVR) to least squares regression to central differences. The *domain* in the legend specifies the blur kernel size in the case of central differences, or the support region over which training examples are gathered for regression. Fig. 4 illustrates the type of imagery on which we evaluated the different methods – animal classes drawn from the ImageNet dataset, often exhibiting large variations in pose, rotation, scale and translation.



(a) (b)

Fig. 2 An experiment to illustrate the alignment performance of pixel intensities versus densely sampled sparse features (in this case densely extracted SIFT descriptors). In both scenarios, we initialize a bounding box within an image, then perturb its location by a given RMS point error (x-axis) and measure the frequency with which the perturbed patch converges back to the initialization (y-axis). In (a) we perform *intra*-image alignment, where the template and perturbation are sampled from the same image. In (b) we perform *inter*-image alignment, where the perturbation is sampled from a different image of the same object class with known ground-truth alignment. The task of inter-image alignment is markedly more difficult, since the two objects being aligned may be experiencing different lightning and pose conditions. The drop in pixel performance is more pronounced than dense SIFT when moving to the harder task.

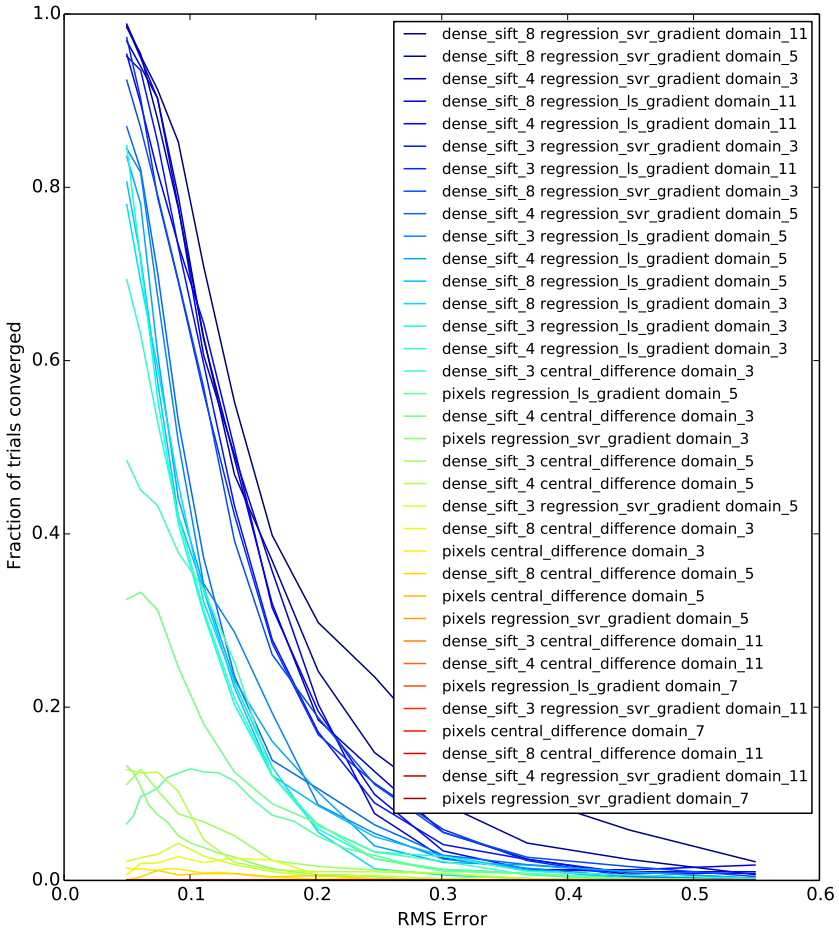


Fig. 3 *Inter*-image alignment performance. We initialize a bounding box within the image, then perturb its location by a given RMS point error (x axis), run Lucas Kanade on the resulting patch, and measure the frequency with which the patch converges back to the initialization (y axis). The *domain* specifies the Gaussian standard deviation in the case of central differences, or the maximum displacement from which training examples are gathered for regression. On dense SIFT, there is a progressive degradation in performance from SVR to least-squares regression to central differences. Pixel intensities (using any gradient representation) perform significantly worse than the top dense SIFT based approaches.

(a) (b)

Fig. 4 Representative pairwise alignments. (a) is the template region of interest, and (b) is the predicted region that best aligns the image to the template. The exemplars shown here all used dense SIFT features and least squares regression to learn the descent directions. The four examples exhibit robustness to changes in pose, rotation, scale and translation, respectively.

Ensemble Alignment

We finish the piece with the challenging real-world application of ensemble alignment. The task of ensemble alignment is to discover the appearance of an object of interest in a corpus of images in an unsupervised manner. Discrete approaches are typically unsuitable to this problem because searching over translation and scale alone is insufficient for good alignment, and exploring higher-dimensional warps using discrete methods is either infeasible or computationally challenging.

We present results using a gradient-based approach called least squares congealing [6]. The details of the algorithm are not essential to our discussion, however it features the same linearization as the LK algorithm, and as such is subject to the same properties we have discussed throughout this chapter.

Fig. 5 show the results of aligning a subset of 170 elephants drawn from the ImageNet dataset,[2] using dense SIFT features and least squares regression, parametrized on a similarity warp. The same set-up using pixel intensities failed to produce any meaningful alignment. Fig. 6 shows the mean of the image stack before and after congealing. Even though individual elephants appear in different poses, the aligned mean clearly elicits an elephant silhouette.

6 Discussion

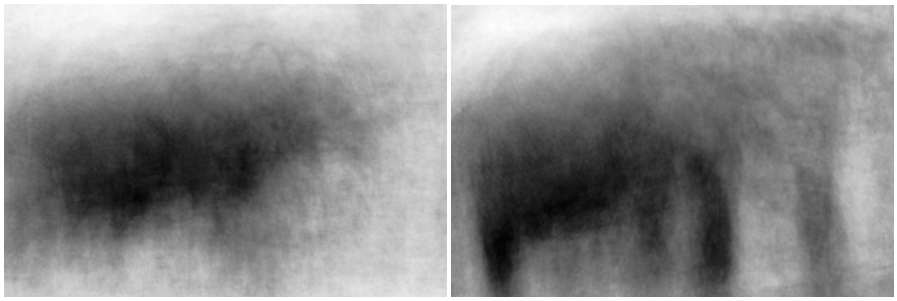
So far in this chapter we have presented the somewhat paradoxical result that densely sampled sparse features perform well in real-world alignment applications (Fig. 2, Fig. 3) whilst sporting poor tangent approximations (Fig. 1). Here we try to offer some insight into why this might be the case.

Consider first the effect of convolving a sparse signal with a low-pass filter. We know from compressive-sensing that observed blurred signals can be recovered almost exactly if the underlying signal is sparse [21]. Unlike traditional dense pixel representations whose high-frequency information is attenuated when convolved with a low-pass filter, sparse signals can be blurred to a much larger extent without any information loss before reaching the limits of sampling theory. Fig. 7 illustrates the effect of comparing dense and sparse signals as the degree of misalignment and blur increases.

The immediate implication of this for image alignment is that a sparse multi-channel representation can be blurred to dilate the convergent region whilst preserving information content. The encoding of local pixel interactions ensures this information content contains high-frequency detail required for good alignment.

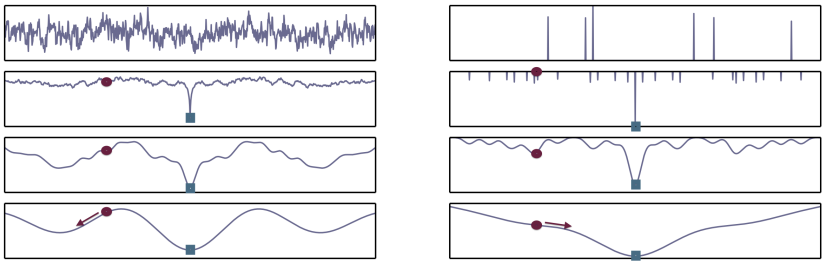
[2] We removed those elephants whose out-of-plane rotation from the mean image could not be reasonably captured by an affine warp. The requirement of a single basis is a known limitation of the congealing algorithm.

Fig. 5 Unsupervised ensemble alignment (congealing) on a set of 170 elephants taken from ImageNet. The objective is to jointly minimize the appearance difference between all of the images in a least-squares sense – no prior appearance or geometric information is used. The first 6 rows present exemplar images from the set that converged. The final row presents a number of failure cases.



(a) (b)

Fig. 6 The mean image of Fig. 5 (a) before alignment, and (b) after alignment with respect to a similarity warp. Although individual elephants undergo different non-rigid deformations, one can make out an elephant silhouette in the aligned mean.



(a) (b)

Fig. 7 A $1D$ alignment thought experiment. The first row shows two signals: a dense signal with a $\frac{1}{f}$ frequency spectrum, and a sparse positive signal. The second, third and fourth rows show the negative auto-correlation of the signals to simulate the expected loss for varying degrees of misalignment (x-axis) with increasing amounts of Gaussian blur applied to the original signals (row-wise). The red circles represent a hypothetical initialization of the parameters (in this case x-translation), the green squares represent the global optima, and the arrows indicate the direction of steepest descent. For the given initialization, gradient-based alignment on the dense signal will never converge to the true optima. Even with a large amount of blur applied, the solution is divergent (the gradient of the cross-correlation is moving away from the optima). The sparse signal, on the other hand, can tolerate a larger amount of blur and still maintain the location of the optima, in this case converging with the greatest amount of blur applied. This illustrates the importance of sparse, positive representations when matching misaligned signals. In order to retain discriminative appearance information, modern features use *multi-channel*, sparse, positive representations – but the basic concept remains.

7 Conclusion

Image alignment is a fundamental problem for many computer vision tasks. In general, there are two approaches to solving for the optimal displacement parameters: (1) to iteratively linearize the image function and take gradient steps over the parameters directly, or (2) to exploit redundancies in the set of allowable deformations and enumerate the set using graphical models. While a large body of research has focussed on gradient-based alignment strategies in the facial domain, they have rarely been applied to broader object categories. For general objects, alignment in pixel space performs poorly because low frequency information in the signal is dominated by lighting variation. Densely sampled sparse features provide tolerance to local image contrast variation, at the expense of reducing the range over which tangent approximations to the image function are accurate. As a result, graphical models have become the preferred approach to alignment when using densely sampled sparse features.

We motivated this chapter with the surprising result that although the tangent approximation is poor, the real-world results when using image features are impressive. We offered some insights into why this may be the case, along with a number of approaches for estimating the descent directions. We ended the piece with an unsupervised ensemble alignment experiment to illustrate how gradient-based methods can operate on challenging imagery with high-dimensional warp functions.

In summary, we showed that the application of gradient-based methods to general object alignment problems is possible when using densely sampled sparse features, and their capacity to handle complex warp/regularization schemes may facilitate some interesting new approaches to existing challenges in image and object alignment.

References

1. S. Avidan. Support vector tracking. *Pattern Analysis and Machine Intelligence (PAMI)*, 26(8):1064–72, Aug. 2004.
2. S. Baker and I. Matthews. Lucas-Kanade 20 Years On: A Unifying Framework: Part 1. *International Journal of Computer Vision (IJCV)*, 56(3):221–255, Feb. 2004.
3. S. Baker and I. Matthews. Lucas-Kanade 20 Years On: A Unifying Framework: Part 2. *International Journal of Computer Vision (IJCV)*, 2004.
4. M. J. Black and A. Jepson. Eigentracking: Robust matching and tracking of articulated objects using a view-based representation. *International Journal of Computer Vision (IJCV)*, 26(1):63–84, 1998.
5. T. Cootes, G. Edwards, and C. Taylor. Active appearance models. *Pattern Analysis and Machine Intelligence (PAMI)*, 2001.
6. M. Cox, S. Sridharan, and S. Lucey. Least-squares congealing for large numbers of images. *International Conference on Computer Vision (ICCV)*, 2009.

7. N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. *International Conference of Computer Vision and Pattern Recognition (CVPR)*, pages 886–893, 2005.
8. N. Dowson and R. Bowden. Mutual information for Lucas-Kanade Tracking (MILK): an inverse compositional formulation. *IEEE transactions on pattern analysis and machine intelligence*, 30(1):180–5, Jan. 2008.
9. H. Drucker, C. Burges, L. Kaufman, A. Smola, and V. Vapnik. Support Vector Regression Machines. *Advances in Neural Information Processing Systems (NIPS)*, (x):155–161, 1997.
10. P. F. Felzenszwalb. Representation and detection of deformable shapes. *Pattern Analysis and Machine Intelligence (PAMI)*, 27(2):208–20, Feb. 2005.
11. P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *Pattern Analysis and Machine Intelligence (PAMI)*, 32(9):1627–45, Sept. 2010.
12. B. K. Horn and B. G. Schunck. Determining optical flow. *Artificial Intelligence*, 17(1-3):185–203, Aug. 1981.
13. C. Liu, J. Yuen, and A. Torralba. SIFT flow: dense correspondence across scenes and its applications. *Pattern Analysis and Machine Intelligence (PAMI)*, 33(5):978–94, May 2011.
14. D. G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision (IJCV)*, 60(2):91–110, Nov. 2004.
15. B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. *International Joint Conference on Artificial Intelligence (IJCAI)*, 1981.
16. L. Rakêt, L. Roholm, M. Nielsen, and F. Lauze. TV-L 1 optical flow for vector valued images. *Energy Minimization Methods in Computer Vision and Pattern Recognition*, pages 1–14, 2011.
17. V. Ramakrishna, D. Munoz, M. Hebert, J. A. Bagnell, and Y. Sheikh. Pose Machines: Articulated Pose Estimation via Inference Machines. *European Conference on Computer Vision (ECCV)*, pages 1–15, 2014.
18. J. Saragih, S. Lucey, and J. Cohn. Face alignment through subspace constrained mean-shifts. *International Conference on Computer Vision (ICCV)*, 2009.
19. J. M. Saragih, S. Lucey, and J. F. Cohn. Deformable Model Fitting by Regularized Landmark Mean-Shift. *International Journal of Computer Vision (IJCV)*, 91(2):200–215, Sept. 2011.
20. E. Simoncelli and B. Olshausen. Natural Image Statistics and Neural Representation. *Annual Review of Neuroscience*, 2001.
21. G. Tsagkatakis, P. Tsakalides, and A. Woiselle. Compressed sensing reconstruction of convolved sparse signals. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014.
22. P. Viola and M. Jones. Robust real-time object detection. *International Journal of Computer Vision*, 2001.
23. J. Weber and J. Malik. Robust computation of optical flow in a multi-scale differential framework. *International Journal of Computer Vision (IJCV)*, 81:67–81, 1995.
24. P. Weinzaepfel, J. Revaud, Z. Harchaoui, and C. Schmid. DeepFlow: Large Displacement Optical Flow with Deep Matching. *International Conference on Computer Vision (ICCV)*, pages 1385–1392, Dec. 2013.
25. X. Xiong and F. De la Torre. Supervised Descent Method and Its Applications to Face Alignment. *International Conference of Computer Vision and Pattern Recognition (CVPR)*, pages 532–539, June 2013.
26. A. Yuille. Deformable templates for face recognition. *Journal of Cognitive Neuroscience*, 3(1):59–70, Jan. 1991.
27. X. Zhu and D. Ramanan. Face detection, pose estimation, and landmark localization in the wild. *International Conference of Computer Vision and Pattern Recognition (CVPR)*, pages 2879–2886, June 2012.
28. K. Zimmermann, J. Matas, and T. Svoboda. Tracking by an optimal sequence of linear predictors. *Pattern Analysis and Machine Intelligence (PAMI)*, 31(4):677–92, Apr. 2009.