Theory and Practice of Globally Optimal Deformation Estimation

Yuandong Tian

Sep 2013

Robotics Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee:
Srinivasa G. Narasimhan, Chair
Martial Hebert
Fernando De La Torre
Kyros Kutulakos, University of Toronto
C. Lawrence Zitnick, Microsoft Research

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy.

Keywords: Image Deformation, Deformable Object, Water Distortion, Document Rectification, Optical Turbulence, Human Pose Estimation, Cloth Deformation, Globally Optimal Solution, Theoretical Analysis, Data-driven Descent, Hierarchical Model, Deep Learning

Abstract

Nonrigid deformation modeling and estimation from images is a technically challenging task due to its nonlinear, nonconvex and high-dimensional nature. Traditional optimization procedures often rely on good initializations and give locally optimal solutions. On the other hand, learning-based methods that directly model the relationship between deformed images and their parameters either cannot handle complicated forms of mapping, or suffer from the Nyquist Limit and the curse of dimensionality due to high degrees of freedom in the deformation space. In particular, to achieve a worst-case guarantee of ϵ error for a deformation with d degrees of freedom, the sample complexity required is $O(1/\epsilon^d)$.

In this thesis, a generative model for deformation is established and analyzed using a unified theoretical framework. Based on the framework, three algorithms, Data-Driven Descent, Top-down and Bottom-up Hierarchical Models, are designed and constructed to solve the generative model. Under Lipschitz conditions that rule out unsolvable cases (e.g., deformation of a blank image), all algorithms achieve *globally optimal* solutions to the specific generative model. The sample complexity of these methods is substantially lower than that of learning-based approaches, which are agnostic to deformation modeling.

To achieve global optimality guarantees with lower sample complexity, the structure embedded in the deformation model is exploited. In particular, Data-driven Descent relates two deformed images that are far away in the parameter space by compositional structures of deformation and reduce the sample complexity to $O(C^d \log 1/\epsilon)$. Top-down Hierarchical Model factorizes the local deformation into patches once the global deformation has been estimated approximately and further reduce the sample complexity to $O(C_1^d + C_2 \log 1/\epsilon)$. Finally, the Bottom-up Hierarchical Model builds representations that are invariant to local deformation. With the representations, the global deformation can be estimated independently of local deformation, reducing the sample complexity to $O(\left(\frac{C}{\epsilon}\right)^{d_0})$ ($d_0 \ll d$). From the analysis, this thesis shows the connections between approaches that are traditionally considered to be of very different nature. New theoretical conjectures on approaches like Deep Learning, are also provided.

In practice, broad applications of the proposed approaches have also been demonstrated to estimate water distortion, air turbulence, cloth deformation and human pose with state-of-the-art results. Some approaches even achieve near real-time performance. Finally, application-dependent physics-based models are built with good performance in document rectification and scene depth recovery in turbulent media.

Acknowledgments

I would like to thank my advisor Srinivasa G. Narasimhan for helping me tremendously with this dessertation. He is a great advisor and teaches me extensively about research, presentation and writing skills over the five years.

I would like to thank my thesis committee members, Martial Hebert, Fernando De La Torre, Kyros Kutulakos and Larry Zitnick, for going through the entire theis and giving me critical comments and advices. Thank all my collaborators, Mohit Gupta, Li Zhang, Dong Huang and Jun Zhu, I learn a lot from your distinctive points of view in solving research problems. Thank my friends for creative discussions, supportive comments and interesting thoughts.

I would like to thank my father and mother for encouraging me to pursue my dreams. Your son is doing great! Finally and most importantly, I would express my love to my dear fiancée, Lufei Ruan, for supporting me extensively over my last year of PhD, during which I was able to build up my thesis story and finish the most part of it. I was so fortunate in Carnegie Mellon and will remember this amazing instutite forever:-)

Thank you all, wholeheartedly!

Contents

Chapter 1

Introduction

In our daily lives, we observe a large range of deforming surfaces or objects such as water, clothing, human body and so on (Fig. 1.1). Modeling and estimating deformation from images can help better understand and hence extract useful information of the scene with these objects. For example, capturing cloth deformation in real time leads to markerless motion capture, and estimating paper deformation helps facilitate digitization of a piece of receipt or a page of book using smart-phone. A famous example is the recent success of Microsoft Kinect that stems from breakthrough on human pose estimation using a depth camera.

Many vision problems deal with complex geometric variations of the same target (e.g., scene or object). Deformation modeling can build the correspondence among different appearances (different images) of the same object, and hence help solve many fundamental vision problems, such as image alignment and image matching, and build compact models for object detection and recognition.

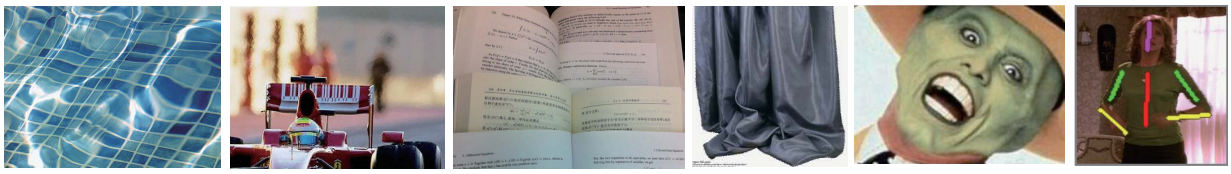


Figure 1.1: Non-rigid image deformations in real-world scenarios caused by water fluctuation, hot air turbulence, book/paper curving, cloth folding, face expression and human pose changes. Modeling and estimating deformation help in extracting useful information from these scenarios.

1.1 Taxonomy of Deformation

Taxonomy of deformation. By definition, the word *deformation* covers a large range of geometric variations and can be categorized according to different criteria. **(1)** Deformation could be *rigid* or *nonrigid*. Rigid deformation preserves certain geometric quantities such as the distance between two locations and angle between two straight lines, and thus has fewer degrees of freedom. Nonrigid deformation has more degrees of freedom and usually maps lines into curves. **(2)** Deformation can either be mainly restricted to a *local* region, such as image deformation of underwater scenes due to small water waves, or can include global motion in addition to local formation, i.e., *non-local* deformation, e.g. a bird flying while flapping its wings.

Note that local distortion means that, each part of the object/scene moves around its original location with small displacement. **(3)** Deformation also be *continuous* that extends smoothly on a surface such as that due to moving a piece of cloth, or *articulated* that has discontinuity on joint locations and object boundary, e.g., human appearance variations due to pose changes. **(4)** Deformation could happen in a 3D volumetric region (e.g., hot air turbulence) or on a 2D surface such as cloth deformation. **(5)** Considering the image formation process, deformation can be created by *dynamic medium* between the camera and the scene, in which the medium property inbetween can bend the light transport and create deformation (e.g., distortion by water surface fluctuation or hot air turbulence), or by the variation of camera internal/external parameters (e.g. perspective distortion caused by unusual camera pose, lens distortion caused by fish-eye lens and so on).

Taxonomy of deformation modeling. A large volume of previous works have been developed to deal with different types of deformation,. **(1)** Researchers have proposed *holistic* approaches that take the entire image as an input, and output an estimation of deformation; Alternatively, *part-based* approaches estimate the location and local deformation of each part of the object/scene, and then combine the evidences together to obtain a globally consistent estimate of deformation. **(2)** Statistically, *generative* approaches establish a model on how the deformation is generated and find the best model parameters using optimization or sampling. *Discriminative* approaches, on the other hand, directly learn a mapping from the input deformed image to the parameter space. **(3)** Finally, deformation modeling could be *specific*, i.e., use significant domain knowledge to facilitate deformation estimation, or is a *generic* approach with mild assumptions that is widely applicable to many scenarios.

1.2 Problem Definition and Technical Challenges

Despite the variety of deformations, they all have one common mathematical formulation. Knowing one template image I_0 and a deformed image I, the goal of this thesis is to find a *deformation field* $W(\mathbf{x})$ that warps the deformed image I back to the template I_0: that:

$$I(W(\mathbf{x})) = I_0(\mathbf{x}) \tag{1.1}$$

Intuitively, for every pixel location $\mathbf{x}$ on the template I_0, $W(\mathbf{x})$ is the corresponding location on the deformed image I. Since deformation could be nonrigid, $W(\mathbf{x})$ is spatially smooth but nonlinear.

Generative (Optimization) point of view. One traditional way to solve Eqn. 1.1 is to minimize the following objective function:

$$J(W) = \sum_{\mathbf{x}} \|I(W(\mathbf{x})) - I_0(\mathbf{x})\| \tag{1.2}$$

which is then regarded as a general optimization problem. Contrary to its simple form, it turns out that finding a globally optimal solution is technically challenging due to the following two reasons.

First of all, appearance created by deformation is image-dependent and is thus **nonlinear** with respect to the deformation parameters. Even for a rigid deformation like 2D translation, the pixel intensity will shift to its nearby pixel value, creating a nonlinearity. A simple local linearization often results in locally optimal solutions or inaccurate prediction.

Coupled with nonlinearity, the difficulty in deformation modeling also stems from the issue of **high degrees of freedom**, or its high dimensionality. For rigid deformation that has only a few number of parameters and low-dimensional (e.g., affine transform with 6 parameters), nonlinearity can be solved by an exhaustive search in the parameter space, i.e., by looking at every parameter and finding the best solution. However, deformation in its general form, in particular nonrigid deformation, is intrinsically high-dimensional. For example, the dimensionality of a deformation field $W(\mathbf{x})$ locally parameterized by a set of image landmarks is roughly twice the number of landmarks. In such a case, exhaustive search could take exponential time to finish, and is usually ineffective and time-consuming. For smooth deformation, dimensional reduction such as PCA or GPLVM [85] can be used. However, in practice, even the reduced dimensionality is still high.

In general, it is very hard to overcome the two technical difficulties within the general optimization framework. Existing methods, such as gradient descent, Newton's method and so on,

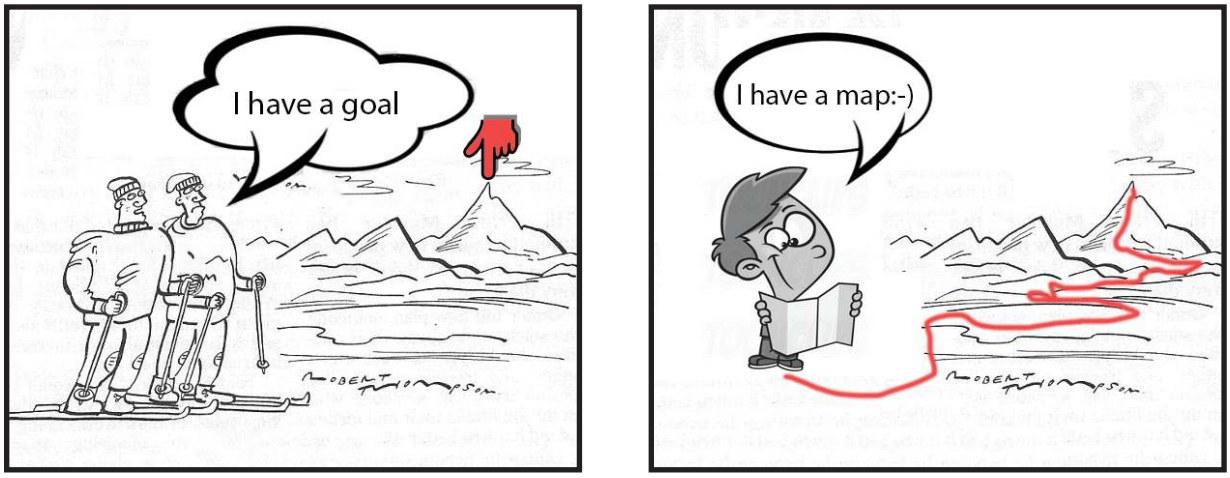


Figure 1.2: General optimization framework versus proposed algorithms in this thesis. **(a)** General optimization sets a goal first, and then goes towards the goal using standard, domain-independent procedure. **(b)** Proposed algorithms directly find a path towards the globally optimal solution without setting the objective.

give numerical procedures that achieve a locally optimal solution and not a global one. To obtain better solutions, good initializations using empirical heuristics are required. In fact, it is highly implausible to find a generic numerical procedure that always returns a guaranteed solution to an arbitrary optimization problem.

So let us take one step back. First, why bother developing generic approaches that solve everything? An algorithm specific to Eqn. 1.1 can potentially work better for our task. Second, why rely on an objective function? Indeed, as show in Fig. 1.2, the objective (Eqn. 1.2) only gives us a goal to aim for but never tells us *how to reach it*. Conversely, if we already know how to reach the goal, why bother setting up a criterion?

Therefore, this thesis instead looks for a specific set of algorithms (or procedures) that directly return a solution to the generative model (Eqn. 1.1) for deformation. The proposed algorithms are specific and may not be broad enough to handle every optimization problem, which is not our goal in any case. The algorithm may not derive from any objective function. The punchline is: the solution obtained by our algorithms has global optimality guarantees.

Discriminative (Predictive) point of view. The other popular way to solve Eqn. 1.1 is to pose it as a prediction task. In this setting, training samples $(I^{(i)}, W^{(i)})$ that contain N deformed images with known deformation fields are generated from the generative model Eqn. 1.1. Then a predictive model is built for the mapping $I \mapsto W$ using the training samples $(I^{(i)}, W^{(i)})$. That

model is applied to a test image I_{test} and returns the most relevant deformation field as the answer.

With this approach, a good answer can always be found if a correct model happens to be picked, together with a sufficient number of training samples. Now the challenging questions become: which model should be used and how many samples are needed? Without knowing the nature of the image-parameter mapping $I \mapsto W$, the predictive model has to be treated as a black-box with the following inevitable fundamental trade-off: simple models such as linear regressors require few samples, with strong assumptions that could be terribly wrong for a highly nonlinear problem like Eqn. 1.1; on the other hand, complicated, nonparametric models could handle arbitrary data distribution but the required number of training samples grows exponentially with respect to dimensionality. This phenomenon, known as the curse of dimensionality (Fig. 1.3(a)), makes predictive models unusable for high-dimensional problems. The recent rapid boost of computational resources may compensate for some exponential growth in practice, but may not suffice to model all deformation.

The only solution is to *find the correct model* by analyzing the problem structure in details. For example, as shown in Fig. 1.3(b), domain knowledge, e.g. the deformation model specified by Eqn. 1.1, implies long-range constraints on the relationship between two faraway regions in the space that are not connected by local smoothness, local factorization (low-rank) structures and so on. In contrast, traditional machine learning approaches aim to find a universal solver that works for arbitrary distributions. The knowledge related to the specific problem is regarded as *features*, an empirical component without formal analysis. As a result, the proposed models in this thesis may not be as straightforward as linear regression or decision forests, but have far better theoretical worst-case guarantees for the specific problem to be solved.

Interestingly and surprisingly, it turns out that the two seemingly orthogonal perspectives, the generative point of view and discriminative points of view, result in a unified framework, which is the main contribution of this thesis. Under this framework, three algorithms are proposed to deal with Eqn. 1.1 with theoretical guarantees. These algorithms follow from the detailed analysis and are built in a principled manner. They assume reasonable Lipschitz conditions that rule out very few unsolvable images, and in many aspects coincide with different heuristics that have been tried to improve the performance of practical systems without knowing the underlying principles.

1.3 Contributions of the Thesis

(a) A novel and principled theoretical framework. This thesis provides a framework that can be used to analyze non-rigid deformation estimation algorithms, including their convergence

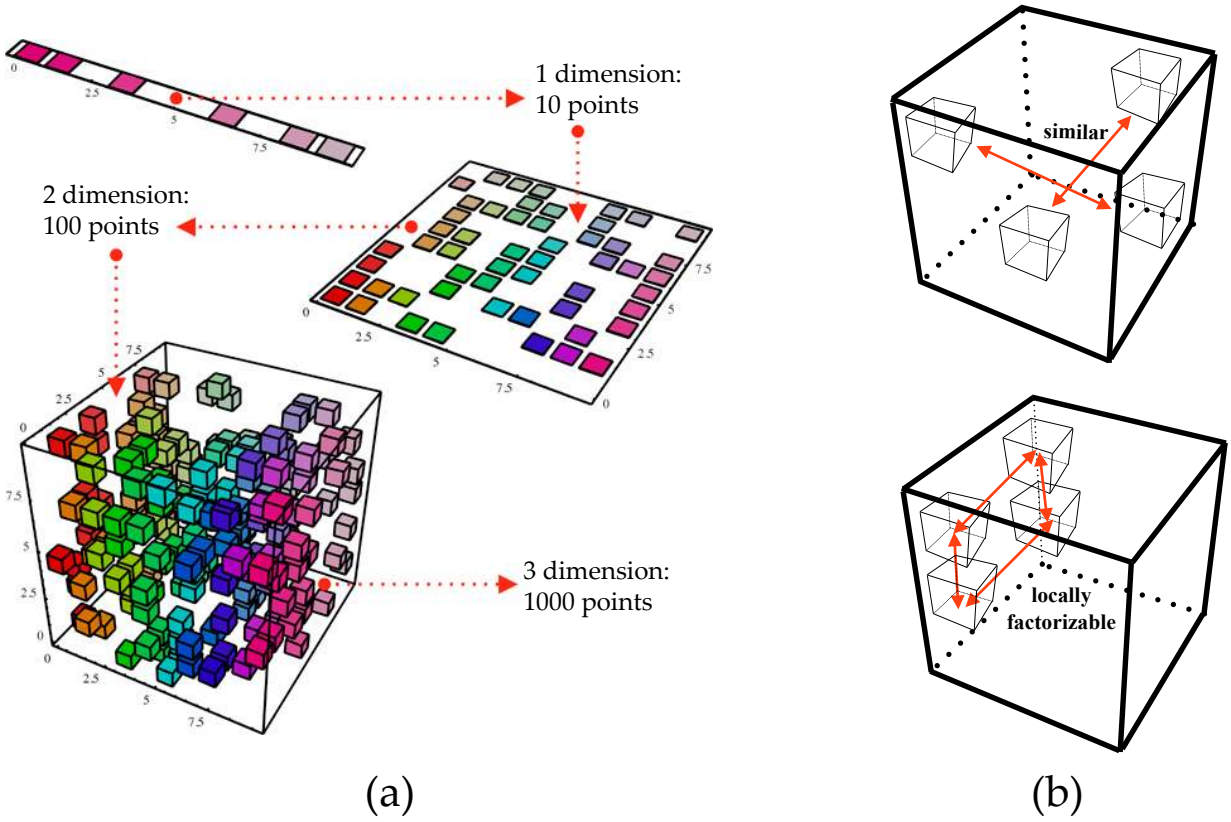


Figure 1.3: The curse of dimensionality and the proposed solution in this thesis. **(a)** The content within the volume goes up exponentially with respect to the dimensionality (degrees of freedom, DoF). As a result, an arbitrary high-dimensional mapping may have exponentially many parameters and an enormous number of training samples are needed to learn it with worst-case guarantees. **(b)** Reducing the degrees of freedom with domain-specific constraints. These constraints may take different forms, e.g., *Top:* there are connections in two regions that are far away in the space, and *Bottom:* Local region of the space may exhibit factorizable (low-rank) structures. Note in our model, these structures are not assumed a prior but follows naturally from the generative model of deformation. With these constraints, the number of parameters used to determine a high-dimensional mapping is substantially reduced. As a result, the sample complexity is also reduced.

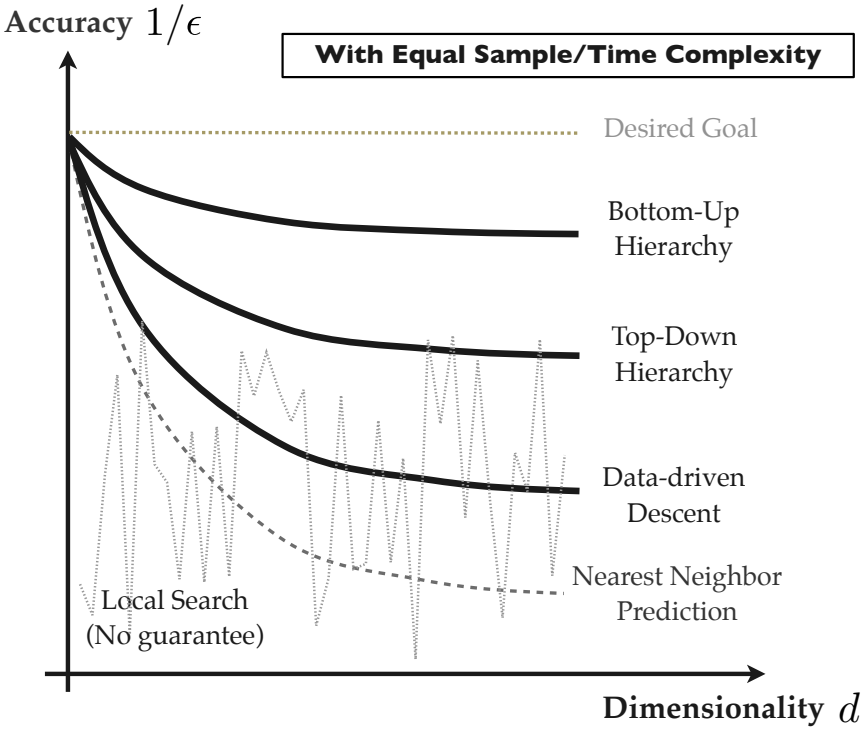


Figure 1.4: Comparison of multiple methods in theoretical performance with the same sample complexity. Local search approaches give a solution without guarantees. Proposed approaches (solid lines) are better than Nearest Neighbor Prediction. d is the dimensionality (degrees of freedom) of deformation and ϵ is the prediction error.

behaviors and their performance guarantees. This framework consists of three steps. First, Lipschitz conditions are assumed for complicated nonlinear and nonparametric relationship between the difference in image appearances and difference in deformation parameters. Second, using Lipschitz conditions, the performance of a single Nearest Neighbor predictor can be guaranteed. Finally, algorithms are constructed by applying the predictors iteratively in a proper way.

The framework has several advantages. First, all algorithms from this framework, by construction, will *never* fail given sufficient training samples (or sufficient time to run). Simple tasks require fewer samples while difficult tasks may need more samples. The number of samples needed is controlled by the constants of Lipschitz conditions that are image-dependent. Second, using sample complexity as a metric, theoretical performances of proposed methods can be measured, independent of their empirical evaluations. Finally, this framework provides direct ways to construct and analyze algorithmic procedures without objective functions.

(b) Three deformation estimation algorithms with global optimality guarantees and reduced sample complexity. Based on the theoretical framework, to solve Eqn. 1.1, this thesis proposes three algorithms, i.e., Data-driven Descent, Top-down and Bottom-up Hierarchical

Method	Sample Complexity
Local Search	—
Nearest Neighbor Prediction	$O(1/\epsilon^d)$
Data-Driven Descent	$O(C^d \log 1/\epsilon)$
Top-Down Hierarchical Prediction	$O(C_1^d + C_2 \log 1/\epsilon)$
Bottom-Up Hierarchical Prediction	$O\left[(C/\epsilon)^{d_0}\right]$ with $d_0 \ll d$

Table 1.1: Comparison of theoretical performance of multiple methods in terms of sample complexity.

Predictions. Under Lipschitz conditions that rule out a few trivial unsolvable cases (e.g., no guarantee could be achieved for a deformed blank image), these methods all have worst-case performance guarantees for deformation estimation, despite the fact that Eqn. 1.1 is nonlinear, nonconvex and high-dimensional with dimensionality d. Furthermore, their performances are differentiated in their required sample complexity to achieve such guarantees, as shown in Tbl. 1.1.

In the first part of the thesis, Data-driven Descent discovers the fact that the two deformed images may be far away from each other in terms of image distance metric, but are closely connected by the compositional structure of deformation. As illustrated in Fig. 1.3(b), such hidden "links" restrict the degrees of freedoms of a high-dimensional deformation space, and reduces sample complexity to $O(C^d \log 1/\epsilon)$. This breaks the Nyquist Limit $O(1/\epsilon^d)$ achieved by general-purpose Nearest Neighbor prediction.

The second part of the thesis develops part-based approaches to further reduce the sample complexity. Top-down Hierarchical Prediction utilizes the (conditional) factorization structure of local deformation, once the global deformation has been estimated correctly. As a result, it achieves a better sample complexity bound of $O(C_1^d + C_2 \log 1/\epsilon)$ by predicting the global deformation first, and localizes the residual deformation onto local patches. Besides, the Lipschitz conditions are also relaxed to handle noise and patch boundaries, yielding a smaller constant C_1 compared to C.

However, the estimation of global deformation has to take care of the image appearance changes caused by both global and local deformation, and still suffers from the curse of dimensionality. To address this problem, Bottom-up Hierarchical Prediction builds representations that are invariant to local deformation in a bottom-up manner. From these representations, the global deformation can be estimated independently of the local one, reducing the dimensionality further to $O((C/\epsilon)^{d_0})$ with $d_0 \ll d$. Once the global deformation is estimated, Top-Down Prediction can follow with little computational cost. This finally breaks the curse of dimensionality. Furthermore, the Lipschitz conditions are only assumed on the smallest patches and will automatically

hold for patches of the higher level whose representations are computed from below. In comparison, Top-Down Hierarchical Prediction assumes the conditions for patches of all levels.

More generally, using the idea of Bottom-Up Hierarchical Prediction, representations of any level can be built by gradually throwing away degrees of freedom that is unrelated to the decisions of that level. As a result, the topmost representations have only a few but discriminative degrees of freedom and can be predicted with substantially fewer number of samples. The analysis can potentially give useful insights into the current deep architectures that have shown promise in empirical studies but lack theoretical interpretations.

(c) New connections between existing methods. From the analysis of the three models, connections are shown between approaches that are traditionally considered to be of very different nature, e.g., the relationship between Lucas-Kanade and Deformable Parts Model, between Generative and Discriminative Models, and between Message Passing and Deep Learning. It also provides new insights into approaches that are not yet fully understood, e.g., Deep Learning.

(d) Broad applications. In addition to deep theoretical analysis, this thesis has also demonstrated broad applications of proposed approaches. Data-Driven Descent has been applied to deformation estimation of water distortion and air turbulence, and has shown accurate landmark localizations. Top-Down Hierarchical Prediction shows not only good accuracy but also near real-time performance (3-4fps), when dealing with water distortion and cloth deformation. Finally, Bottom-up Hierarchical Prediction has been applied to Human Pose Estimation, showing state-of-the-art performance and natural-looking human pose samples.

(e) Application-specific modeling. Besides the framework, this thesis also builds physics-based deformation models for document images captured by cellphone cameras, and remote scene appearances under hot air turbulence. Using these models, OCR-friendly document images can be recovered and scene depth can be estimated with good performance.

Chapter 2

Related work

Nonrigid deformation has long been a research topic that attracts much attention. Depending on different statistical characteristics, the methods of deformation estimation also vary. One typical deformation is continuous, e.g., a video sequence of a scene under fluctuating water, an object printed onto a bending rubber-sheet, or a piece of textured cloth moving with the wind. Usually there is no background clutter and every captured pixel is a deformed version of the scene/object under consideration. Thus, no detection is needed. Besides, there could be self-occlusion or not.

The other typical deformation is the one with articulated object, such as human body. Human body could depict a variety of poses and show substantial self-occlusion. In addition, human may occupy a small portion of the image, while the remaining region of the image is the background clutter that may easily trigger false detection.

In the following, we will list the related works for both cases.

2.1 Estimation of continuous deformation

There has been a long and rich history of studying geometric transformations between two images, assuming both images contain the same object or the scene. To list them all is beyond the scope of this paper. Here we only discuss the works that are most relevant to this thesis.

Traditionally, the image without deformation is called the template, or the reference image, while the other image is the deformed image. Most of the approaches use holistic approach to model the deformation, rather than partition the image into parts.

Generative Approaches. In this framework, one starts with the process of how the deformation is generated, and search over the parameter space to find the (locally) best parameter that matches with the observed deformed image. This is often formulated as a minimization problem, whose minimizer are the desired set of parameters.

In particular, for deformation estimation, starting from the classical optical flow algorithm by Lucas and Kanade [54], these approaches minimize the function $J(\tilde{\mathbf{p}}) = \|I_{\mathbf{p}} - I_{\tilde{\mathbf{p}}}\|^2$ with respect to the parameter $\tilde{\mathbf{p}}$. The intensity difference between the distorted template $I_{\tilde{\mathbf{p}}}$ under the current parameter estimate $\tilde{\mathbf{p}}$ and the test image $I_{\mathbf{p}}$, is iteratively minimized until it reaches a local minimum.

Under the same minimization framework, many successive works achieve faster convergence by using a constant Hessian matrix. As a trade-off, restrictions on the type of warping have to be placed. For example, the forward compositional approach [88] requires the warping to be compositional. The inverse additive method [30] requires the warping to be separable or spatially linear. Inverse compositional approaches [5, 57, 102] require the warping to be both compositional and invertible. These conditions restrict the possible applications of these methods. Other methods, including Active Appearance Models [16, 57], Direct Appearance Models [36] and Difference Decomposition [29, 76] are applicable to a wide class of distortions and are fast. However, it is not clear which function is minimized and there is no guarantee for convergence.

Free-form medical image registration [69] adopts a multilevel approach in which distortion parameterized by a B-spline is optimized to align two images at each level. The resulting estimated distortion is nonparametric and hence no predefined types of warping are required. But the algorithm may still be trapped within local optima. Recently, to address this problem, a convex approximation to the objective function has been learned [62, 106], but whether it remains faithful under large distortions is unclear.

A Markov Random Field can also be used to model image deformation [82], but the underlying combinatorial problem is NP-hard and approximate inference techniques, such as linear programming relaxation or Tree-reweighted Message Passing, have to be used to obtain a solution without guarantee.

Discriminative Approaches. This research direction starts from the idea of learning a direct mapping from the distorted image to the template, based on a training set with known distortion parameters. The simplest example is the nearest-neighbor approach, while more advanced approaches can also be used. However, although such kind of approaches do not suffer from local minima, it has its own problems. If the parameters are high-dimensional, then either a certain parameteric form of the regression has to be assumed, or the number of training samples needed is exponential with respect to the dimensionality. One way to address this problem is to find a low-dimensional representation (called "latent variables") of the parameter space, e.g. using PCA or GPLVM[85]. Then the prediction is made in the low-dimensional space. To alleviate dimensionality problem, recent works [15] uses a "successive regression" technique for non-rigid image alignment. They first estimate the coarse parameters and use the coarse parameters to

extract details features from the image, and do a regression again. However, their approaches are heuristic and do not have a theoretical guarantee. As a proposed work, we aim to find a similar algorithm with guarantees.

Combining discriminative and generative approaches. Since both generative and discriminative approaches have their advantages and disadvantages, there have been many attempts to combine both. One popular strategy [75, 89] is to first find a coarse estimation using the discriminative approach. Then, using this estimation as the initialization, a generative method is applied in the second stage for refinement. This requires that the first prediction be sufficiently close to the global optimum. Randomly generated training samples are also used in the iterative procedure, e.g. Hyperplane Approximation [43], which is similar in spirit to what we proposed in Chapter 4. However, they use a spatially linear distortion model along with a linear estimator (hyperplane) that does not guarantee global optimality. Also they do not relate the distribution of random training samples to the convergence of the algorithm. In Rosales and Sclaroff [68], from candidate predictions made by multiple predictors, a generative model is used to choose the best one as the final output.

Feature-based Approaches. The third research direction uses highly distinctive local features for sparse matching, e.g. SIFT [53]. Being rotation and scale invariant, such local features can be used to match images with large viewpoint changes, under analytic transformations such as affine or perspective, and with occlusions. Salzmann and Fua [74] also use such local features to find the point correspondences in the case of non-rigid deformation, but trustworthy local matches are sparse and spatial models have to be included to obtain denser correspondences [34, 116].

Pattern tracing. Finally, for some deformable object that has known periodic textures or patterns, e.g. an image with pure text content, one can find the lattice and build the deformation in this way. [52] uses auto-correlation of image to discover the location of patterns and analyzes the symmetric group structure using crystallographic group theory. [32] discovers the periodic and deformed patterns from a local region of the image in a greedy manner by iteratively proposing new unit and assigning neighbors between the pattern units. [63] formulates pattern tracing as a multi-target tracking problem in the spatial domain and greedily grow the discovered patterns in two directions. In Chapter 9, we face a slightly different setting that along the text line the patterns (characters) can be different, also across the text line the line spacing can be different but still smooth-changing. We estimate the deformation by tracing the textlines (patterns) along one direction, and then collecting the curve together to obtain the entire deformation field.

The representation of deformation can either be parametric (e.g. cubic spline [21] and thin-plate spline [63]) or nonparametric (e.g. Chapter 9 and [111]). Parametric methods use a fixed number of parameters to characterize the entire curve. They are more robust to noise and oc-

clusion, yet for complicated deformation they may not capture fine structures. Nonparametric methods adapt the number of its parameters to the complexity of the curve (e.g., parameterizing the curve using control points with fixed spacing). They are more flexible, can handle arbitrary deformation, but are more sensitive to noise.

2.2 Estimation of Articulated Objects

For articulated object (mainly human pose), depending on whether a clutter background is present, previous works take different approaches.

Clean background. In this case, most of the previous works take the regression-based (discriminative) approach. Some methods treat the image in a holistic manner. The feature are silhouette [2], sparse SIFT-like features [119], Haar-like wavelet [9] and so on. The regressor includes Relevant Vector Regression [2], Gaussian Processes [119], Boosting [9], Mixture of Experts [90], Random Forests [66]. Some can model ambiguity of pose predictions and deal with one-to-many mapping [96]. Other approaches take the part-based approach and learn the part detector only. One famous example is the Microsoft Kinect [86] that trains each part of human body separately using Random Decision Forest. No spatial model is used.

Cluttered background. In this case, simple regression can not work well due to unexpected image content from background. Also it is no longer a valid assumption that the image mostly contains a single object. Thus the majority of work focus on the following two-stage framework. train a part detector and use a spatial model to combine the local detection consistently. Since part detectors are intrinsically ambiguous, It is critical to design the spatial model so as to capture a versatile yet plausible set of poses.

The influential work of Felzenszwalb et al. [22] uses pictorial structures (PS) [24] to efficiently capture the pairwise spatial relationships between nearby parts. The resulting structure forms a tree allowing for efficient inference. A disadvantage of [22] is that it only allows for small deformations from a fixed template. To solve this problem, [39] used a (global) mixture of pictorial structures to capture greater variations in pose. However, since the number of plausible human poses is exponential, the number of parameters that need to be estimated is prohibitive without a large dataset or part sharing mechanism.

Recently, [113] treats each part rather than the entire body as a mixture of templates while modeling their pairwise relationships. As a result, it offers a compact way to represent an exponential number of poses with shared parameters. Unfortunately, their use of pairwise relationships fails to capture the complex characteristics of pose space. Other works [8, 97, 107] model the high-order relationship among parts with high-order cliques. However, either their inference

is approximate, or a heuristic search is used with worse-case exponential time complexity.

As first proposed by David Marr [56], one way to introduce high-order relationships without losing the benefit of efficient tree-based inference is to build a hierarchical structure with latent nodes. Both [91] and Chapter 11 in this thesis follow this paradigm and use latent nodes with part mixtures from [113]. The difference is that in the settings of [113], each mixture component of a latent node also corresponds to one HoG template, modeling the image appearance covered by the descendants of that node. We will see detailed comparison in Chapter 11.

2.3 3D Reconstruction from deformation

Given image deformation, one direct application is to find the 3D configuration of the scene/object that causes it. This involves a 3D lifting from 2D projections, an ill-posed problem that needs additional physical constraints.

In modeling cloth deformation, several methods have been proposed to reconstruct the 3D shape of cloth. One constraint is the inextensibility of the clothing. That is, two points on the cloth must be of the same length in the reconstruction. [73] shows a close-form solution of 3D structure can be obtained with this constraint using eigen-decomposition. [70] relax the inextensibility constraint to be inequality and results in a convex formulation for 3D reconstruction. Both are optimization-based approaches that assume sparse correspondences between reference image and current frame are known via SIFT. Recently, [80] jointly formulates the point correspondence problem and 3D reconstruction by minimizing an energy function and [71] uses a regressor to predict the locations of correspondence and refine the prediction by a local optimization. In all works, they assume perspective camera.

In document reconstruction, many previous works assume a strong global shape model of the document, e.g., part of a cylinder [13], piecewise cylinder [118] or a developable surface [50, 51, 117]. In Chapter 9, in contrast, we assume a local prior that is reasonable for document reconstruction and enforce no global shape.

Other approaches, called shape-from-texture, start by estimating the local differential quantities that the 3D shape projects onto the captured image, e.g. projected tangent angle [110], texture distortion [55] and foreshortening [25], and then collect them to form a global 3D structure. Since they all minimize non-linear objective functions, the estimation is not guaranteed to produce the global optimum [25]. In Chapter 9, we formulate shape-from-texture in the specific context of text document images as a homogeneous least square problem, in which the globally optimal solution can be obtained using SVD. Previous work [19] also uses SVD for 3D reconstruction of a scene given a set of curves as intersection between the scene and a set of planes.

From their setting, all points on the same curve must lie on the same plane.

2.4 Hierarchical Models

Hierarchical structures have been used extensively in vision. Typical scenarios include coarse-to-fine optimization [69] for a better local solution, interest point detection [53], multi-resolutional feature extraction [49], biologically plausible framework for object recognition [78] and so on. Recently, it is also used in Deep Learning, showing state-of-art performance in image classification [48]. However, as far as we know, none of the previous works provide theoretical performance guarantees.

2.5 Deformation caused by medium

Deformation can also be caused by medium property between the camera and the scene. In such a case, deformation usually takes the form of local distortion in which each pixel moves around its true location over time.

Here we consider two different cases that cause distortion, both due to a change of refractive index. One typical case is that there exists an interface between two media (e.g. water and air), and light transport is altered through the interface, causing distortion. The other typical case is that a stochastic process is ongoing over the volumetric medium itself, causing a fluctuating field of refractive index.

Firstly, undistortion is possible if the exact shape of the interface can be measured or estimated. In [42], the interface and the scene are illuminated by spectrally isolated red and green channels, and captured by a color camera. The surface shape is thus measured by one channel and used to undistort the other. In [59], a known scene (calibration pattern) is tracked to estimate the surface. This thesis follows such an idea, but with a data-driven approach rather than relying on potentially time-consuming calibration. In Chapter 4, I show shape estimation of an interface between air and water from a video sequence or two images.

Secondly, if the goal is to undistort the scene but not reconstruction, averaging can be used. Given a video sequence of a distorted scene, simple pixel-wise mean/median works well for reducing small fluctuations[81]. A better approach is to select only "good" image patches from the video frames and stitch them together (also known as the "lucky image"[27]). Several works [17, 18, 20] find the center of the distribution of patches from the video as the undistorted patch, either by embedding them on a manifold[20] or by clustering them[17, 18]. Wen et al [108]

model the frames as random phase perturbations in the Fourier domain, and average them using the bispectrum to undistort the image.

Thirdly, the scene information can also be extracted from the *variance* of the location of moving scene points embedded in the medium over time. One such example is the optical turbulence due to random fluctuations of temperature gradients near warm surfaces. The turbulence will cause the shimmering and distortion of the scene. Previous works in remote sensing and astronomical imaging on turbulence largely treat turbulence as a negative effect and focus on how to correct images through turbulence. For atmospheric turbulence, the distorted wavefronts arriving from stars can be optically corrected using precisely controlled deforming mirror surfaces, beyond the angular resolution limit of telescopes [67]. For terrestrial imaging applications, recent works have proposed to digitally post-process the captured images to correct for distortions and to deblur images [28, 31, 35, 41, 122]. Optical-flow based methods have been used further to register the image sequences to achieve modest super-resolution [84]. Compared to these works, in Chapter 10, we show from Kolmogorov's seminal works [44] that there exists a relationship between the depth of a scene point and the variance of its location in the image plane over time, and hence turbulence can be used to estimate the scene depth.

Part I

Theory I: Data-driven Predictions of Deformation

Chapter 3

Mathematical Modeling of Deformation

3.1 Image Formation

Given a template image I_0 and a vector of parameters $\mathbf{p}$, a distorted image $I_\mathbf{p}$ is computed using a *generating function* G:

$$I_\mathbf{p} = G(I_0, \mathbf{p}) \tag{3.1}$$

In particular, the template is at the origin of the parameter space, i.e., $I_0 = G(I_0, 0)$. We denote $\mathcal{I}$ as the manifold that consists of all possible distorted images that can be generated from Eqn. (3.1):

$$\mathcal{I} = \left\{ I_\mathbf{p} = G(I_0, \mathbf{p}) \;\middle|\; \forall \mathbf{p} \in \mathbb{R}^d \right\} \tag{3.2}$$

The function G can be implemented using an *image warp* $W(\mathbf{x}; \mathbf{p})$ that maps a pixel $\mathbf{x}$ to the position $W(\mathbf{x}, \mathbf{p})$. The warp $W(\mathbf{x}; \mathbf{p})$ is controlled by a set of parameters $\mathbf{p}$. Typically $W(\mathbf{x}, 0) = \mathbf{x}$. The warp $W(\mathbf{x}, \mathbf{p})$ can be applied to the template in either forward or backward direction:

$$G_\mathrm{F}(I_0, \mathbf{p}): \quad I_\mathbf{p}(W(\mathbf{x}; \mathbf{p})) = I_0(\mathbf{x}) \tag{3.3}$$

$$G_\mathrm{B}(I_0, \mathbf{p}): \quad I_\mathbf{p}(\mathbf{x}) = I_0(W(\mathbf{x}; \mathbf{p})) \tag{3.4}$$

Intuitively, the forward generating function pushes every pixel $\mathbf{x}$ in the template to the location $W(\mathbf{x}; \mathbf{p})$ in the distorted image, while the backward generating function pulls the pixel located at $W(\mathbf{x}; \mathbf{p})$ of the template back to the location $\mathbf{x}$ of the distorted image. A particular family of distortions may satisfy either Eqn. (3.3) or Eqn. (3.4), but not necessarily both. For invertible warpings, both representations are equally valid. To make things simple, throughout the thesis, we will use Eqn. 3.3 as the generative model.

For now, we will focus on occlusion-free warps in the 2D image space. For out-of-plane

Images	
I_0	Template image.
I	An arbitrary 2D Image.
$\mathbf{x}, x, y$	2D pixel location, $\mathbf{x} = (x, y)$ is the vector form, while x and y are components.
Parameters	
$\mathbf{p}$	All parameters as a vector.
$\mathbf{p}(j)$	j-th component of parameter $\mathbf{p}$.
$I_{\mathbf{p}}$	A deformed image with ground truth deformation parameter $\mathbf{p}$.
$W(\mathbf{x}; \mathbf{p})$	Deformation field parametrized by $\mathbf{p}$
$B(\mathbf{x})$	Deformation bases, each column is a basis function.
d	Effective degrees of freedom
D	Number of warp bases.
K	Number of landmarks.
G	Generative function that takes parameters and returns an deformed image.
$\mathcal{I}$	Deformation manifold. Collection of all deformed images under generative model G.
H	Pull-back operation
Lipschitz Conditions	
r	Range of the parameters.
L_1, L_2	Lipschitz Constants.
ΔI	Difference between two images I_1 and I_2, $\Delta I = \|I_1 - I_2\|$
$\Delta\mathbf{p}$	Difference between two parameters $\mathbf{p}_1$ and $\mathbf{p}_2$, $\Delta\mathbf{p} = \|\mathbf{p}_1 - \mathbf{p}_2\|_\infty$.
Samples and Algorithms	
N	Number of training samples.
T	Number of iterations.
$(\mathbf{p}^{(i)}, I^{(i)})$	A training pair. A deformed image $I^{(i)}$ and its ground truth parameter $\mathbf{p}^{(i)}$.
$\{\mathbf{p}^{(i)}, I^{(i)}\}_{i=1}^N$	A collection of training samples.
ϵ	Prediction error. Inverse of prediction accuracy.
β	Inverse of sample density.
γ	(One minus) convergence rate.
C	Constant terms in big-O notations.
Miscellaneous	
$\lceil x \rceil$	Ceiling function, $\lceil x \rceil$ is the smallest integer that is greater than or equals to x.
$\|\mathbf{p}\|_\infty$	Infinite norm. $\|\mathbf{p}\|_\infty = \max_i \|\mathbf{p}(i)\|$.

Table 3.1: Notations used in this thesis.

rotation or deformation in the presence of occlusion, it is possible that $I_{\mathbf{p}}(W(\mathbf{x};\mathbf{p})) \neq I_0(\mathbf{x})$ even if $\mathbf{p}$ is the correct parameter, since some information in the template has been discarded due to deformation.

3.2 Parameterization of Warping

The warp $W(\mathbf{x};\mathbf{p})$ could be high-dimensional. In the extreme case, $\mathbf{p}$ could encode 2-dimensional displacement for every pixel, yielding a $2mn$ dimensional vector for an m-by-n template. For any 200-by-200 image, the length of $\mathbf{p}$ would be 80000 and is too large to handle.

Fortunately, the effective degrees of freedom is not so high since nearby pixel displacements have to be correlated. Therefore, we assume the warping $W(\mathbf{x})$ is on a low dimensional subspace and can be represented as follows:

$$W(\mathbf{x};\mathbf{p}) = \mathbf{x} + B(\mathbf{x})\mathbf{p} \tag{3.5}$$

where $B(\mathbf{x})$ is a set of *warp bases* that can be obtained *a priori* using either analytic models or measured data or complex physical simulations. In the context of forward warping, Eqn. 3.3 means for every pixel located at $\mathbf{x}$ in the template, $W(\mathbf{x};\mathbf{p})$ is the corresponding 2D location in the deformed image $I_{\mathbf{p}}$.

The bases function $B_G(\mathbf{x}) = [\mathbf{b}_1(\mathbf{x}), \ldots, \mathbf{b}_D(\mathbf{x})]$ is a 2-by-D matrix, and the parameters $\mathbf{p}_G$ is a D-by-1 vector. Given $\mathbf{x}$, each basis function $\mathbf{b}_k(\mathbf{x})$ outputs a 2-dimensional column vector $[\mathbf{b}_k^x(\mathbf{x}), \mathbf{b}_k^y(\mathbf{x})]^\top$, representing the displacement at location $\mathbf{x}$. $B(\mathbf{x})$ can capture spatially nonlinear distortions and thus covers a broad range of distortions, including affine transform, Thin-Plate Spline (TPS), lens distortion, water distortion and changes of facial expressions [57], and cloth deformation [73].

If $B(\mathbf{x})$ is orthogonal, which means for two bases $\mathbf{b}_j(\mathbf{x})$ and $\mathbf{b}_k(\mathbf{x})$:

$$\int \mathbf{b}_j^x(\mathbf{x})\mathbf{b}_k^x(\mathbf{x})\mathrm{d}\mathbf{x} + \int \mathbf{b}_j^y(\mathbf{x})\mathbf{b}_k^y(\mathbf{x})\mathrm{d}\mathbf{x} = 0 \quad \forall j \neq k \tag{3.6}$$

then each basis $\mathbf{b}_k(\mathbf{x})$ contributes to exactly one degree of freedom, and therefore, the *effective degrees of freedom* d of deformation field $W(\mathbf{x};\mathbf{p})$ is the number D of bases. Since warping is the only reason the deformed image changes its appearance, d (or D) is also the degrees of freedom of deformed image $I_{\mathbf{p}}$. If the bases in $B(\mathbf{x})$ are *overcomplete* and is not orthogonal (or even not independent), then the effective degrees of freedom $d < D$. In such a case, one can reduce D by a Gram-Schmidt orthogonalization process. However, in some cases, it is more

Name	Dimensionality	Analytical forms
Pure Translation	2	$B(\mathbf{x}) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$
Affine bases	6	$B(\mathbf{x}) = \begin{bmatrix} 1 & 0 & x & x & y & y \\ 0 & 1 & y & -y & x & -x \end{bmatrix}$
Len Distortion	4-6	$\mathbf{b}_k^x(\mathbf{x}) = \mathbf{b}_k^y(\mathbf{x}) = \|\mathbf{x} - \mathbf{x}_0\|^k$.
Thin-Plate Spline (TPS)	2#Landmark	$\mathbf{b}_{2k}^x(\mathbf{x}) = \mathbf{b}_{2k+1}^y(\mathbf{x}) = \|\mathbf{x} - l_k\| \log \|\mathbf{x} - l_k\|$
Radial bases function (RBF)	2#Landmark	$\mathbf{b}_{2k}^x(\mathbf{x}) = \mathbf{b}_{2k+1}^y(\mathbf{x}) = \exp(-\frac{\|\mathbf{x}-\mathbf{l}_k\|^2}{2\sigma^2})$
Facial Deformation [57]	#Bases	Bases learnt from training data

Table 3.2: $\mathbf{l}_k$ is the k-th landmark location

convenient to work on an overcomplete set of bases.

Invertible versus Non-invertible warping. Some warps following Eqn. 3.5 are invertible, e.g., affine warps, while others are not invertible. Tbl. 3.2 shows a list of useful bases. As a special case, the 8-dimensional perspective warps $W_{\text{persp}}(\mathbf{x}; \mathbf{p})$:

$$W_{\text{persp}}(\mathbf{x}; \mathbf{p}) = \mathbf{x} + \left[\frac{p_1 x + p_2 y + p_3}{p_7 x + p_8 y + 1}, \frac{p_4 x + p_5 y + p_6}{p_7 x + p_8 y + 1} \right] \qquad \mathbf{x} = (x, y) \tag{3.7}$$

cannot be represented using Eqn. 3.5, but it is an invertible warp.

3.3 The Pull-back Operation

Given an arbitrary image I and a parameter $\mathbf{q}$, one defines the *pull-back* image $H(I, \mathbf{q})(\mathbf{x})$ as follows:

$$H(I, \mathbf{q})(\mathbf{x}) \equiv I(W(\mathbf{x}; \mathbf{q})) \tag{3.8}$$

Intuitively, instead of retrieving a pixel at $\mathbf{x}$, we retrieve the pixel at $W(\mathbf{x}; \mathbf{q})$ from I and place it at $\mathbf{x}$ on the pulled-back image. From the definition, obviously we have $H(I, 0) = I$.

In the special case that I is generated from a (usually unknown) parameter $\mathbf{p}$, for compactness, I also define the notation $I_{\mathbf{p};\mathbf{q}} \equiv H(I_{\mathbf{p}}, \mathbf{q})$ as the pull-back operation. From definition, we have

$$H(I_{\mathbf{p}}, \mathbf{p}) = I_0 \tag{3.9}$$

by Eqn. 3.3. More generally, we have the following *pull-back inequality*:

$$\|H(I_{\mathbf{p}}, \mathbf{q})(\mathbf{x}) - I_{\mathbf{p}-\mathbf{q}}(\mathbf{x})\| \leq c_H \|\mathbf{p} - \mathbf{q}\| \qquad \forall \mathbf{x} \tag{3.10}$$

Intuitively, the operation H approximately "cancels out" the deformation caused by parameter $\mathbf{p}$ with a parameter $\mathbf{q}$, yielding a possibly less distorted image $I_{\mathbf{p}-\mathbf{q}}$.

In the specific case of invertible warping, e.g., affine, perspective, we can simply define H as the inverse warp:

$$H(I, \mathbf{q}) \equiv G(I, \mathbf{q}^{-1}) \tag{3.11}$$

where $\mathbf{q}^{-1}$ is the parameters of the inverse warp. In this case, $c_H \equiv 0$.

3.4 Estimation of Deformation Parameters $\mathbf{p}$

The main focus of this thesis, also the main task of deformation estimation, is to find the optimal parameters $\mathbf{p}$, if $I_{\mathbf{p}}$, I_0 and G (or warping function W) are known.

Estimation of $\mathbf{p}$ from labeled correspondences. If correspondences are known, then under the model 3.5, estimating $\mathbf{p}$ is simple. Given a set of N points $\{\mathbf{x}_i\}$ on the template and their correspondences $\{\mathbf{x}'_i\}$, finding the optimal parameters $\mathbf{p}$ is straightforward:

$$E'(\mathbf{p}) = \min_{\mathbf{p}} \sum_i \|\mathbf{x}'_i - (\mathbf{x}_i + B(\mathbf{x}_i)\mathbf{p})\|^2 \tag{3.12}$$

which can be solved analytically:

$$\hat{\mathbf{p}} = (B^{xT}B^x + B^{yT}B^y)^{-1}[B^{xT}\delta\mathbf{x} + B^{yT}\delta\mathbf{y}] \tag{3.13}$$

where $\delta\mathbf{x} = [x'_1 - x_1, x'_2 - x_2, \ldots, x'_N - x_N]^T$, $\delta\mathbf{y} = [y'_1 - y_1, y'_2 - y_2, \ldots, y'_N - y_N]^T$ are both N-by-1 vectors. $B^x = [B^x(\mathbf{x}_1); B^x(\mathbf{x}_2); \ldots; B^x(\mathbf{x}_N)]$ and $B^y = [B^y(\mathbf{x}_1); B^y(\mathbf{x}_2); \ldots; B^y(\mathbf{x}_N)]$ are both N-by-K matrices.

However, it is usually very hard to estimate $\mathbf{p}$ without knowing the correspondences.

3.5 Generative Approach

A typical generative view regarding to Eqn. 3.3 is to assume random (Gaussian) noise on top of $I_{\mathbf{p}}$ and to find a prediction $\hat{\mathbf{p}}$ that minimizes the following nonconvex objective under the principle of Maximum Likelihood:

$$\hat{\mathbf{p}} = \arg\min_{\hat{\mathbf{p}}} E(\hat{\mathbf{p}}) \equiv \arg\min_{\hat{\mathbf{p}}} \|I_{\mathbf{p}}(W(\mathbf{x}, \hat{\mathbf{p}})) - I_0(\mathbf{x})\|_2^2 \tag{3.14}$$

The seminar work, Lucas-Kanade[6], starts from the current estimation $\hat{\mathbf{p}}$ and locally *lin-*

earizes Eqn. 3.14 under a small parameter increment:

$$E(\hat{\mathbf{p}} + \delta\hat{\mathbf{p}}) = \|I_{\mathbf{p}}(W(\mathbf{x}, \hat{\mathbf{p}} + \delta\hat{\mathbf{p}})) - I_0(\mathbf{x})\|_2^2 \tag{3.15}$$

$$\approx \|I_{\mathbf{p}}(W(\mathbf{x}, \hat{\mathbf{p}})) + \nabla I_{\mathbf{p}}(W(\mathbf{x}; \hat{\mathbf{p}}))^\top B(\mathbf{x})\delta\hat{\mathbf{p}} - I_0(\mathbf{x})\|_2^2 \tag{3.16}$$

For mathematical simplicity, denote $I_{\mathbf{p};\hat{\mathbf{p}}} \equiv H(I_{\mathbf{p}}, \hat{\mathbf{p}}) = I_{\mathbf{p}}(W(\cdot; \hat{\mathbf{p}}))$. Solving for $\delta\hat{\mathbf{p}}$ in a least square manner and we obtain:

$$\delta\hat{\mathbf{p}} = -Q^{-1}\delta f \tag{3.17}$$

where

$$Q(\hat{\mathbf{p}}) = \int B^\top(\mathbf{x})\nabla I_{\mathbf{p}:\hat{\mathbf{p}}}(\mathbf{x})\nabla I_{\mathbf{p}:\hat{\mathbf{p}}}^\top(\mathbf{x})B(\mathbf{x})\mathrm{d}\mathbf{x} \tag{3.18}$$

and

$$\delta f(\hat{\mathbf{p}}) = \int \nabla I_{\mathbf{p}:\hat{\mathbf{p}}}^\top(\mathbf{x})B(\mathbf{x})\left[I_{\mathbf{p}:\hat{\mathbf{p}}}(\mathbf{x}) - I_0(\mathbf{x})\right]\mathrm{d}\mathbf{x} \tag{3.19}$$

The algorithm is listed in Alg. 1.

Algorithm 1 Lucas-Kanade Algorithm for Deformation Estimation

1: **INPUT:** Test image I_{test}. Initial guess $\hat{\mathbf{p}}^0$.
2: **for** $t = 1 : T$ **do**
3: Compute the pull-back image $I^t = H(I_{\text{test}}, \hat{\mathbf{p}}^t)$.
4: Compute the increment $\delta\hat{\mathbf{p}}^t = -Q_t^{-1}\delta f_m$, where

$$Q_t = \int B^\top(\mathbf{x})\nabla I^t(\mathbf{x})(\nabla I^t(\mathbf{x}))^\top B(\mathbf{x})\mathrm{d}\mathbf{x} \tag{3.20}$$

and

$$\delta f_t = \int (\nabla I^t(\mathbf{x}))^\top B(\mathbf{x})\left[I^t(\mathbf{x}) - I_0(\mathbf{x})\right]\mathrm{d}\mathbf{x} \tag{3.21}$$

5: Set $\hat{\mathbf{p}}^{t+1} = \hat{\mathbf{p}}^t + \delta\hat{\mathbf{p}}^t$
6: **end for**
7: **return** $\hat{\mathbf{p}}^{M+1}$ as the final estimation $\hat{\mathbf{p}}$.

Unfortunately, unless the template image $I_0(\mathbf{x})$ is of some special forms, Eqn. 3.14 is a nonconvex and nonlinear optimization problem and local search approach, e.g., Alg. 1 can only give a local optimal solution.

Multiple tricks have been proposed to increase the convergence range of Alg. 1. This includes multiple initial guess, blurring the test image I_{test} for smoother gradient computation, and building an image pyramid for a coarse-to-fine search. Empirically, these heuristics work great but there is no guarantee that $\hat{\mathbf{p}}$ will achieve the ground truth $\mathbf{p}$ within the energy minimization

framework Eqn. 3.14.

3.6 Discriminative Approach

On the other hand, from the discriminative view, Eqn. 3.3 represents a (possibly nonlinear) relationship between $\mathbf{p}$ and $I_{\mathbf{p}}$. Therefore, by learning such a relationship with labeled training samples $\{\mathbf{p}^{(i)}, I_{\mathbf{p}^{(i)}}\}$, one can give a prediction from an unseen test images I_{test} that is also generated from the same model. More importantly, from the discriminative point of view, it is possible we can have a principled framework to analyze iterative procedures like Alg. 1 globally that is yet not possible within the generative framework.

In this thesis, *the three proposed algorithms are all rooted from this observation*.

3.6.1 The Lipschitz Condition

Since we do not know the specific parametric form of mapping, to avoid any model bias, it is better to assume a nonparametric model, such as a Nearest Neighbor predictor: for an image I, find $I_{\mathbf{p}^{(i)}}$ in the training set that is closest to I, and return the parameter $\mathbf{p}^{(i)}$ as the prediction. Although other nonparametric predictive models, such as decision tree, boosting, support vector regression, can also be used and may practically achieve better performance, in this thesis we focus on theoretical analysis on nearest neighbor prediction due to its simplicity.

A basic assumption for such an approach to work, is to assume there is a positive correlation between the image difference $\Delta I \equiv \|I_{\mathbf{p}_1} - I_{\mathbf{p}_2}\|$ in terms of a certain image metric (e.g., l^2 norm) and the parameter difference $\Delta\mathbf{p} \equiv \|\mathbf{p}_1 - \mathbf{p}_2\|_\infty$ in terms of maximal parameter differences. Intuitively, this means that if two images are close, so are their parameters and vice versa, which can be represented by the following *Lipschitz conditions*:

$$L_1 \Delta I \leq \Delta\mathbf{p} \leq L_2 \Delta I \qquad \forall \mathbf{p}_1, \mathbf{p}_2 : \|\mathbf{p}_1\|_\infty, \|\mathbf{p}_2\|_\infty \leq r_0 \tag{3.22}$$

where, $0 < L_1 < L_2 < +\infty$ are two positive constants that are dependent on the template image I_0. $r_0 > 0$ is the range for the condition to hold. The larger r_0 is, the more strict the conditions will be. Without loss of generality, L_1 and L_2 are assumed to be the tightest bounds.

Note that $L_1 = 0$ is the case where two distinct images share the same parameters, and $L_2 = +\infty$ is the multi-valued mapping case in which a single image is associated with multiple parameters. In both cases, the Eqn. 3.22 does not hold. As we shall see, the sample complexity goes to $+\infty$.

Throughout this thesis, the Lipschitz conditions defined by Eqn. 3.22, as well as all its extensions, are *assumptions*. This means that they may hold for some images, but may not hold for other images. An obvious failure case is that the template I_0 is uniform and $\Delta I \equiv 0$ whatever Δp is. However, we emphasize that for most images, such conditions hold and the proposed analysis works. The goal of the analysis, is to find the weakest conditions so that the worst-case guarantees will hold.

3.7 Nearest Neighbor Prediction and The Nyquist Limit

Under the Lipschitz assumption (Eqn. 3.22), we now are able to show the first theoretical bound for Nearest Neighbor predictors, as demonstrated in Thm. 3.7.1.

Theorem 3.7.1 (Sample Complexity for Nearest Neighbor predictor) *If Eqn. 3.22 holds, then with*

$$\left\lceil \frac{L_2}{L_1}\frac{r_0}{\epsilon} \right\rceil^d \tag{3.23}$$

number of training samples, for an image $I_{\mathbf{p}}$ generated from Eqn. 3.3 with $\|\mathbf{p}\|_\infty \leq r_0$, the predictor is able to make a prediction $\hat{\mathbf{p}}$ that is ϵ-close to the true parameter $\mathbf{p}$:

$$\|\hat{\mathbf{p}} - \mathbf{p}\|_\infty \leq \epsilon \tag{3.24}$$

Proof By Theorem 13.1.2, we can uniformly sample the hypercube $[-r_0, r_0]^d$ in the parameter space so that for any $\|\mathbf{p}\|_\infty \leq r_0$, there exists $(\mathbf{p}^{(i)}, I^{(i)})$ so that

$$\|\mathbf{p} - \mathbf{p}^{(i)}\|_\infty = \max_j |\mathbf{p}(j) - \mathbf{p}^{(i)}(j)| \leq \frac{L_1}{L_2}\epsilon \tag{3.25}$$

From the same theorem, the number of samples needed is exactly Eqn. 3.23. On the other hand, a Nearest Neighbor prediction $(\mathbf{p}_{NN}, I_{NN})$ can only be closer to the test $I_{\mathbf{p}}$ in the image space. Then by Lipschitz condition, we have:

$$\|I_{NN} - I_{\mathbf{p}}\| \leq \|I^{(i)} - I_{\mathbf{p}}\| \leq \frac{1}{L_1}\|\mathbf{p}^{(i)} - \mathbf{p}\|_\infty \leq \frac{1}{L_2}\epsilon \tag{3.26}$$

Again use Lipschitz condition and notice $\mathbf{p}_{NN}$ is the output prediction, we have:

$$\|\hat{\mathbf{p}} - \mathbf{p}\|_\infty = \|\mathbf{p}_{NN} - \mathbf{p}\|_\infty \leq L_2\|I_{NN} - I_{\mathbf{p}}\| \leq \epsilon \tag{3.27}$$

∎

Intuitively, Thm. 3.7.1 shows that the ratio $L_2/\epsilon L_1$ is the samples complexity per dimension for guaranteed Nearest Neighbor prediction. For simple images that contain one salient object with a clear background, $L_2/\epsilon L_1$ is small and a few samples suffice; while for difficult images with cluttered background or repetitive patterns, $L_2/\epsilon L_1$ is large and a lot of samples are needed to distinguish among locally similar-looking structures.

3.8 The Nyquist Limit

Although we define Eqn. 3.22 and present Thm. 3.7.1 in the context of image deformation, they actually can be used to characterize any input/output mapping and Thm. 3.7.1 will still hold.

In fact, without exploiting any domain-specific knowledge, $O((1/\epsilon)^d)$ is the best thing we can do. A substantial reduction of training samples is impossible. The intuition is that the mapping $\mathbf{p} \mapsto I_{\mathbf{p}}$, although locally smooth as required by the Lipschitz condition, could be arbitrary over the long range in the high-dimensional space. To cope with such flexibility, one needs to place densely training samples at every location of the space, which naturally leads to the factor $(1/\epsilon)^d$. Therefore, we call $O((1/\epsilon)^d)$ the *Nyquist Limit* since it reflects the least amount of information needed to completely encode the high-dimensional mapping. Note this limit does not rely on specific nonparametric prediction algorithms. So for arbitrarily mapping $\mathbf{p} \mapsto I_{\mathbf{p}}$, if Nearest Neighbor does not work, neither does boosting or any other approaches.

As we shall see, the only way to break the Nyquist Limit is to get domain-knowledge involved. The domain-knowledge gives extra information, in particular, the structural information to the mapping function and substantially reduces the number of samples needed to achieve the same guarantees.

Many previous works have indeed done so. However, their additional assumptions, such as factorization, linearity, convexity, low-rank, sparsity, are more for mathematical (algebraic) convenience rather than for the specific property of domain. As a result, their model may oversimplify the problem and lose important and interesting aspects.

Chapter 4

Data-Driven Descent: Breaking the Nyquist Limit

In this chapter, a novel data-driven algorithm is developed to estimate the deformation in a globally optimal manner. The term "global optimality" means any prediction $\hat{\mathbf{p}}$ provided by the algorithm is ϵ-close to the true parameter $\mathbf{p}$: $\|\hat{\mathbf{p}} - \mathbf{p}\|_\infty \leq \epsilon$. At the same time, the number of samples, if properly distributed, is substantially reduced from $O(1/\epsilon^d)$ to $O(C^d \log 1/\epsilon)$. This sample complexity grows *logarithmically* with respect to the accuracy $1/\epsilon$ (Note C is independent of ϵ.). More importantly, the dimension d is decoupled from the accuracy $1/\epsilon$, breaking the Nyquist Limit.

The proposed algorithm adopts an iterative strategy that successively warps back the distorted test image I towards the template I_0 until convergence. The direction of warping back is guided by training samples, thus we call this method *Data-Driven Descent (DDD)*.

Data-Driven Descent can be applied to a broad class of 2D image distortions demonstrated in Eqn. 3.5. This includes affine warps, and more complex spatially nonlinear distortion (e.g. water and cloth deformation). In particular, it does not require the warping family to form a group, hence has fewer restrictions than previous works [5, 30, 57, 102] that use a similar "warp-back" strategy.

Good performance of Data-Driven Descent has been shown in both synthetic and real experiments with complex deformations due to water fluctuation and cloth deformation, outperforming several existing methods [69, 98].

4.1 The Intuition

We start from a concrete scenario to explain the intuition behind the algorithm and how it excels Nearest Neighbor prediction. Imagine a spaceship that wishes to return to the Earth. However, for some reason the navigation system is faulty and does not know the coordinates of the Earth relative to the current position. Fortunately, there are satellites around the Earth. Each satellite broadcasts a signal containing its coordinates, which can be received by the spaceship.

A straightforward way to localize the spaceship is to find the strongest signal from the closest satellite, and treat the received coordinates as its own. This is the well-known nearest-neighbor approach. The accuracy of such approaches heavily depends on how close the nearest satellite is to the spaceship, or, the local density of satellites.

However, a *fundamentally* different and more efficient way would be to drive the spaceship to another part of the space by the amount of displacement that *sends its nearest satellite back to the Earth*. If satellites are reasonably dense, then the spaceship should go closer to the earth. The spaceship can now receive new information at the new location, find the nearest satellite again and continue to move accordingly. With a proper distribution of satellites, the spaceship can land on the Earth. The original location of the spaceship can be estimated as the summation of all the consecutive readings of the coordinates.

Let us briefly analyze this approach. Obviously, this approach is beyond Nearest Neighbor since it uses satellites that are far from each other, instead of just a nearby cluster. Hence, it requires only a sparse distribution of satellites around the original location of the spaceship, but a dense distribution near the Earth. That is, a coarse estimation suffices to bring the spaceship to the portion of the space with more satellites, where the estimation can be further refined. As a result, using fewer satellites can achieve the same accuracy as compared to Nearest Neighbor approach.

4.1.1 The Algorithm

For images, the story is similar. The only difference is to replace the Earth with the template, the satellites with the training images (samples) and the spaceship with the distorted test image. As illustrated in Fig. 4.1, we start with the distorted test image I and distorted training pairs $\{\mathbf{p}^{(i)}, I^{(i)}\}$. In each iteration k the algorithm finds the closest training image $(\mathbf{p}^{(k)}, I^{(k)})$ to the distorted image I^k in terms of *image metric* and performs a pull-back operation H using $\mathbf{p}^{(k)}$ to obtain a new image I^{k+1}, that is *less distorted* compared to I^k and is *closer* to the template image I_0 in the parameter space. Then, the training sample nearest to I^{k+1} is found, the parameter estimation is updated and the procedure is iterated until the desired accuracy $1/\epsilon$ is obtained, i.e.,

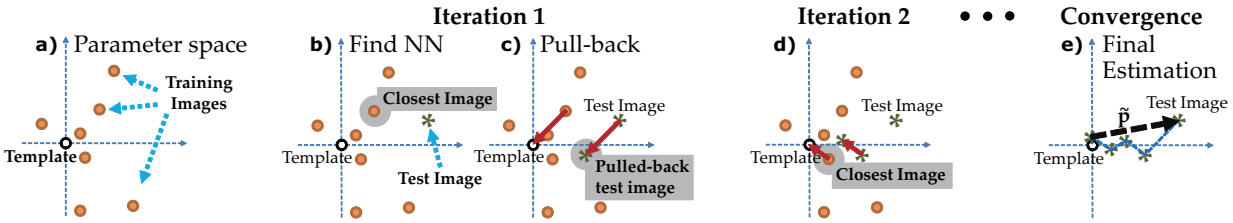


Figure 4.1: Algorithm for distortion estimation. **(a)** The template (origin) I_0 and distorted training images with known parameters $\{\mathbf{p}^{(i)}, I^{(i)}\}_{i=1}^{N}$ are shown in the parameter space. **(b)** Given a distorted test image, its nearest training image $(\mathbf{p}^{(k)}, I^{(k)})$ is found. **(c)** The test image is "pulled-back" using $\mathbf{p}^{(k)}$ to yield a new test image, which is closer to the template than the original one. **(d)** Step (b) and (c) are iterated, taking the test image closer and closer to the template. **(e)** The final estimate $\tilde{\mathbf{p}}$ is the summation of estimations in each iteration.

the estimation is ϵ-close to the template. Finally, the estimate of the distortion parameter $\mathbf{p}$ is given by the cumulative estimation $\tilde{\mathbf{p}}^K$. This algorithm is listed below.

Algorithm 2 The algorithm for distortion estimation

1: **INPUT** The training pairs $\{\mathbf{p}^{(k)}, I^{(k)}\}$. The test image I_{test}.
2: **for** $k = 0 : K$ **do**
3: 1. Find I^k's nearest training image $I^{(k)}$ with known parameter $\mathbf{p}^{(k)}$ i.e., $I^{(k)} = \arg\min_i \|I^k - I^{(i)}\|$.
4: 2. Set cumulative estimation $\tilde{\mathbf{p}}^k = \sum_{j=0}^{k} \mathbf{p}^{(j)}$.
5: 3. Set pulled-back test image $I^{k+1} = H(I_{\text{test}}, \tilde{\mathbf{p}}^k)$. (See Eqn. 3.8 and Eqn. 3.11)
6: **end for**
7: **OUTPUT** $\tilde{\mathbf{p}}^K$ is the final estimation.

To alleviate the possible error accumulation with successive resampling (interpolation), we obtain I^k by pulling-back the original test image I_{test} using the cumulative estimation $\tilde{\mathbf{p}}^{k-1} \equiv \sum_{j=0}^{k-1} \mathbf{p}^{(j)}$ in each iteration.

In the following, we will give a proof of global convergence for Alg. 2 under mild conditions. The idea of the proof is to show after each iteration the norm of residue always shrinks by a constant factor, and thus converges to zero. In other words, it is a coarse-to-fine strategy in the parameter space.

To keep the intuition clear, we start with the family of invertible warps. In this case, H is just the inverse operator of the generating function G. This operator partially cancels out the distortion in $I_{\mathbf{p}}$ by an amount of $\mathbf{q}$, yielding a new distorted image $I_{\mathbf{p}-\mathbf{q}}$ that remains on the

distortion manifold $\mathcal{I}$ (Eqn. (3.2)). This substantially simplifies our analysis. Then we generalize the conclusion to non-invertible warps that take the form of Eqn. (3.5).

4.2 Global Optimality Guarantees

In this section, we prove under the generative model specified in Eqn. 3.5 and under the assumption of Lipschitz condition (Eqn. 3.22), Alg. 2 converges to the global optimum if the training samples are properly distributed.

The warping family (Eqn. 3.5) includes invertible warps that form a group, such as affine and projective transforms [5, 29, 102], and noninvertible warps that satisfy the pull-back inequality (Eqn. 3.8). We also give an upper bound on the number of training samples as a sufficient condition to instantiate this distribution.

4.2.1 The distribution of training samples

Let us consider the set of distorted images whose distortion parameters $\mathbf{p}$ are within the hypercube $S_{r_0} = \{I_{\mathbf{p}}, \|\mathbf{p}\|_\infty \leq r_0\}$. The origin of this space corresponds to the undistorted template image I_0.

We want the training samples to distribute densely near the origin (template) and sparsely at the periphery of the parameter space. Mathematically, given a distorted image $I \in \mathcal{I}$ generated from the distortion model whose parameters satisfy $\|\mathbf{p}\|_\infty \leq r$, we assume that there always exists a training image $I^{(i)} \in \mathcal{I}$ so that:

$$\|I - I^{(i)}\| \leq \frac{\beta r}{L_2} \tag{4.1}$$

where $\beta < 1$. Eqn. (4.1) shows the density decays when moving away from the template to the peripheral of the parameter space (increasing r). With this condition, the following theorem shows Alg. 2 always yields a global optimum estimation for *any* test distorted images within S_{r_0}.

Theorem 4.2.1 (The global convergence of Alg. 2) *If the Lipschitz condition (Eqn. 3.22) holds with constant L_1 and L_2, the pull-back inequality (Eqn. 3.8) holds with constant c_H, the training samples density satisfy Eqn. 4.1 with inverse density β, and $\gamma \equiv 2c_H L_2 + \beta < 1$, then for any test image I_{test} with ground truth parameter $\mathbf{p}$, Alg. 2 computes an estimation $\tilde{\mathbf{p}}^K = \sum_{k=0}^{K} \mathbf{p}^{(k)}$ so that for $\|\mathbf{p}\|_\infty \leq r_0$:*

$$\|\tilde{\mathbf{p}}^K - \mathbf{p}\|_\infty \leq \gamma^{K+1} r_0 \tag{4.2}$$

where $1 - \gamma$ is the rate of convergence. In particular, $\tilde{\mathbf{p}}^K \to \mathbf{p}$ if $K \to +\infty$.

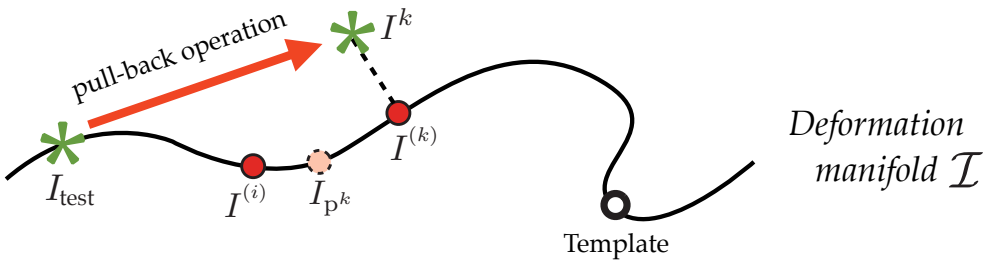


Figure 4.2: Illustration of Theorem 4.2.1.

Proof The intuition of the proof is similar to the spaceship metaphor. With the density condition (Eqn. (4.1)) and the right-hand side of the Lipschitz condition (Eqn. (3.22)), it is guaranteed that in iteration k, the Nearest Neighbor prediction error is always γ times smaller to the norm of parameter residue. As a result, the norm of such difference goes down exponentially with k and the algorithm converges to the true distortion parameters.

In iteration k, The estimation residual is $\mathbf{p}^k \equiv \mathbf{p} - \tilde{\mathbf{p}}^{k-1} = \mathbf{p} - \sum_{j=0}^{k-1} \mathbf{p}^{(j)}$, and particularly $\mathbf{p}^0 = \mathbf{p}$.

From the premise, we have $\|\mathbf{p}^0\|_\infty \leq r_0$. In the following, we prove by induction that the norm of the residue $\|\mathbf{p}^k\|_\infty \leq r_k \equiv \gamma^k r_0$ for any k, and the convergence follows.

Assume those conditions hold for k, in the following we prove they also hold for $k+1$. By the pull-back inequality (Eqn. (3.8)), we have for $I^k = H(I_{\text{test}}, \tilde{\mathbf{p}}^{k-1})$:

$$\|I^k - I_{\mathbf{p}^k}\| \leq c_H \|\mathbf{p}^k\|_\infty \tag{4.3}$$

From the dense condition Eqn. (4.1), there exists a training sample $I^{(i)} \in \mathcal{I}$ that is close to $I_{\mathbf{p}^k} \in \mathcal{I}$:

$$\|I_{\mathbf{p}^k} - I^{(i)}\| \leq \frac{\beta \|\mathbf{p}^k\|_\infty}{L_2} \tag{4.4}$$

Using triangle inequality in the image space, the training sample $I^{(i)}$ is also closer to the rectified image I^k:

$$\|I^k - I^{(i)}\| \leq \left(c_H + \frac{\beta}{L_2}\right) \|\mathbf{p}^k\|_\infty \tag{4.5}$$

Thus, the Nearest Neighbor pair $(\mathbf{p}^{(k)}, I^{(k)})$ of I^k must be even closer:

$$\|I^k - I^{(k)}\| \leq \|I^k - I^{(i)}\| \leq \left(c_H + \frac{\beta}{L_2}\right) \|\mathbf{p}^k\|_\infty \tag{4.6}$$

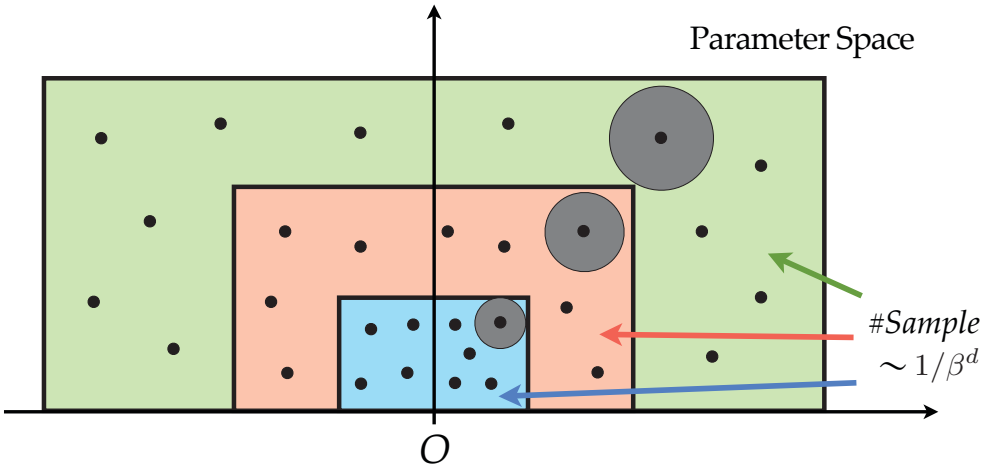


Figure 4.3: The number of samples needed to fill a given hypercube $\|\mathbf{p}\|_\infty \leq r$ is independent of r since the allowed prediction uncertainty (shown in gray solid circle) is proportional to r. As a result, only a small neighborhood of the origin O (the template) requires dense sampling. This is the key to decouple the accuracy from the dimension of the parameter space, which is not attainable for the nearest-neighbor and regression-based approaches.

Combining Eqn. 4.3 and Eqn. 4.6, we obtain an upperbound of the distance of two images $I_{\mathbf{p}^k}$ and $I^{(k)}$, both lying on the manifold $\mathcal{I}$:

$$\|I_{\mathbf{p}^k} - I^{(k)}\| \leq \|I_{\mathbf{p}^k} - I^k\| + \|I^k - I^{(k)}\| \leq \left(2c_H + \frac{\beta}{L_2}\right) \|\mathbf{p}^k\|_\infty \tag{4.7}$$

Thus their parameter is also close according to Eqn. (3.22):

$$\|\mathbf{p}^k - \mathbf{p}^{(k)}\|_\infty \leq (2c_H L_2 + \beta)\|\mathbf{p}^k\|_\infty = \gamma \|\mathbf{p}^k\|_\infty \tag{4.8}$$

which means the difference of current residue $\mathbf{p}^k$ and its estimation $\mathbf{p}^{(k)}$ is bounded by $\gamma\|\mathbf{p}^k\|_\infty$. Note such difference $\mathbf{p}^k - \mathbf{p}^{(k)}$ is precisely the residue $\mathbf{p}^{k+1}$ in the next iteration. By the induction hypothesis, we have:

$$\|\mathbf{p}^{k+1}\|_\infty \leq \gamma\|\mathbf{p}^k\|_\infty \leq \gamma^2\|\mathbf{p}^{k-1}\|_\infty \leq \ldots \leq \gamma^{k+1} r_0 \to 0 \tag{4.9}$$

∎

We verify that $\gamma < 1$ on synthetic data in Section 4.5.2. In contrast, as shown in Thm. 3.7.1, in the Nearest Neighbor case, the training images have to be distributed uniformly in the parameter space to achieve optimal performance for any test sample distributions.

4.3 The Number of Training Samples Needed

An interesting question is how many samples are needed to satisfy the density condition (Eqn. 4.1). We now show the number N of required training images grows only logarithmically with respect to the prediction accuracy $1/\epsilon$.

To start, we now present a sufficient condition for Eqn. (4.1) to hold:

Lemma 4.3.1 *For a given radius r, if the hypercube $\|\mathbf{p}\|_\infty \leq r$ can be covered by a set of smaller hypercubes with side $2(L_1/L_2)\beta r$, then Eqn.* (4.1) *holds.*

Proof If we could achieve this covering, then for any $I \in \mathcal{I}$ and its parameter $\mathbf{p}$ with $\|\mathbf{p}\|_\infty \leq r$, there exists at least one hypercube centered at $\mathbf{p}^{(i)}$ so that

$$\|\mathbf{p} - \mathbf{p}^{(i)}\|_\infty \leq \frac{L_1}{L_2}\beta r \tag{4.10}$$

using the left-hand side of Eqn. (3.22), we have:

$$\|I - I^{(i)}\| \leq \frac{\beta}{L_2} r \tag{4.11}$$

which matches the condition of Eqn. (4.1). ∎

Now let us consider how many small hypercubes (or essentially, the training samples) are required for hypercube-covering in Lemma 4.3.1. Use Thm. 13.1.2, it turns out that the following number of samples suffices to satisfy the condition of Lemma 4.3.1:

$$\left\lceil \frac{r_1}{r_2} \right\rceil^d = \left\lceil \frac{L_2}{\beta L_1} \right\rceil^d \tag{4.12}$$

Crucially, this is independent of r (See Fig. 4.3). Thus, if Alg. 2 terminates in K iterations, $K\lceil L_2/\beta L_1 \rceil^d$ samples would suffice.

On the other hand, using Eqn. (4.2), we can compute $K = \lceil \log(r_0/\epsilon)/\log(1/\gamma) \rceil$ for a given accuracy $1/\epsilon$. As a result, the total number $N(\epsilon, \beta, c_H, L_1, L_2)$ of training images that is sufficient to make Alg. 2 converge to the true parameters (global optimum) is the following:

$$N(\epsilon, \beta, c_H, L_1, L_2) = \left\lceil \frac{L_2}{\beta L_1} \right\rceil^d \left\lceil \frac{\log r_0/\epsilon}{\log 1/\gamma} \right\rceil \tag{4.13}$$

A large β implies fewer training samples in each iteration but requires more iterations to achieve the same accuracy, and vice versa. The optimal β^*, which is independent of ϵ, can be obtained by minimizing Eqn. (4.13).

Note for any L_1 and L_2 that satisfy Eqn. (3.22), following the same reasoning, we conclude the number of training samples is bounded above by Eqn. (4.13). The tightest bound is given by largest L_1 and smallest L_2 satisfying Eqn. (3.22).

As a result, Eqn. (4.13) grows logarithmically with respect to the accuracy $1/\epsilon$. In contrast, as shown in Thm. 3.7.1, Nearest Neighbor requires $\lceil L_2/\epsilon L_1 \rceil^d$ samples for the same accuracy. In Fig. 4.5(b), we show the significant differences in performance between the two methods on synthetic data. Intuitively, the existence of a generating function G substantially restricts the degree of freedom of its inverse mapping. Thanks to this, we can establish the inverse mapping at a good accuracy using significantly fewer samples.

4.4 Possible Extensions to Algorithm 2

Using features. Instead of the raw image I, one can also use features $\phi(I)$ for Nearest Neighbor search. In this situation, L_1 and L_2 are defined between the feature space and the parameter space:

$$L_1\|\phi(I) - \phi(I')\| \leq \|\mathbf{p} - \mathbf{p}'\| \leq L_2\|\phi(I) - \phi(I')\| \tag{4.14}$$

With this definition, Theorem 4.2.1 still holds. A good image feature corresponds to a smaller ratio of L_2/L_1. This means that the feature metric is more correlated to the parameter metric. If they are perfectly correlated ($L_1 = L_2$), then fewest training samples are required.

Using generative approaches as the second stage. When the parameter estimation is very close to the true value, one could use a generative approach to save samples without being trapped into local optima. In such a case, Algorithm 2 can be regarded as a discriminative approach that gives a good initialization.

K_{NN} **nearest-neighbors.** In practice, due to the constant factor $(L_2/\beta L_1)^d$, the N given by Eqn. (4.13) can still be a large number. In this situation, using K_{NN} nearest-neighbors with weighted voting (i.e., kernel regression) can further reduce the required samples, as shown in Fig. 4.5(e).

Fast nearest-neighbors. For N training samples and K iterations, the time complexity of a naïve implementation of Algorithm 2 is $O(NK)$. Currently it takes 5 seconds for a rectification of 300 by 300 image with $N = 1000$ training samples and $K = 20$ iterations using our unoptimized Matlab codes on a Pentium Core 2 machine with a single core. However, many methods used in retrieving approximate nearest-neighbors, such as locality sensitive hashing (LSH), can be applied to reduce the complexity substantially.

Incorporating temporal knowledge. Although Algorithm 2 does not assume temporal rela-

Figure 4.4: Some template images used in synthetic experiments. (See Section 4.5)

tionship between two distorted images, when dealing with distorted video sequence, temporal continuity can be easily incorporated as follows: after the parameter $\tilde{\mathbf{p}}_t$ of the current frame I_t is estimated, we add a new *synthetic* training pair $(\tilde{\mathbf{p}}_t, I_{\tilde{\mathbf{p}}_t})$ to the training set and proceed with the next frame I_{t+1}. If $\tilde{\mathbf{p}}_t$ is an accurate estimation, then I_{t+1} is similar to $I_{\tilde{\mathbf{p}}_t}$ by temporal continuity and will be pulled-back directly near the origin (template) in just one step. If $\tilde{\mathbf{p}}_t$ is not accurate, adding a perfectly labeled training pair will not hurt the performance of the algorithm and does not cause drifting that often occurs in frame-to-frame tracking approaches.

Active training samples. It is possible to include new training images using the generating function G *after* the test image is known. The temporal continuity described above is an example. More generally, the parameters $\tilde{\mathbf{p}}$ estimated by any regression-based method (e.g., Relevant Vector Regression [2] or Gaussian Processes [119]), associated with the synthetic image $I_{\tilde{\mathbf{p}}}$ can be used as a training pair. Multiple regressors may also be used. Then, our algorithm simply selects the one closest to the test in the image metric. Note this is similar in spirit to [68] in which multiple regressors are used for candidate predictions which are then verified by a generative approach.

4.5 Analysis of the algorithm using simulations

4.5.1 Data synthesis

In order to verify the properties of our algorithm, we perform synthetic experiments where the true distortion parameters are known. We simulated distortions on 100 randomly selected images, some of which are shown in Fig. 4.4. The warps are of the form given by Eqn. (3.5), where $B(\mathbf{x})$ are composed of $d = 20$ orthonormal bases computed by applying PCA on randomly generated smooth deformation fields by Gaussian Processes. For each of the 100 template images, we synthesize $N = 1000$ distorted images for the training set and 10 for the test set. Note that a total of 1000 test samples are involved in the simulation and should be sufficient to justify

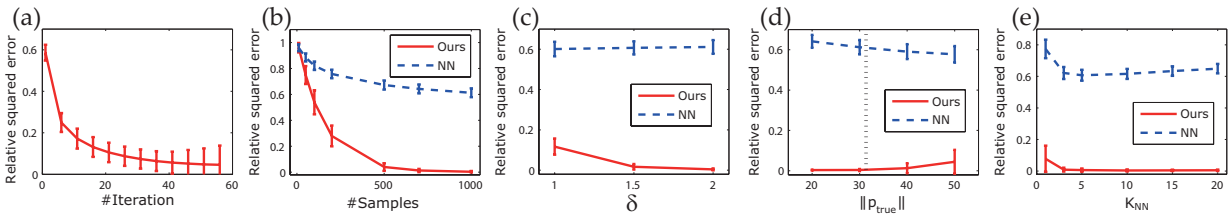


Figure 4.5: The effects of four different factors on the performance of the algorithm in terms of relative squared error $\|\mathbf{p}_{\text{true}} - \tilde{\mathbf{p}}\|^2/\|\mathbf{p}_{\text{true}}\|^2$. **(a)** Average convergence behavior computed over all test images. **(b)** The more training images, the better the performance. Note our method performs much better than nearest-neighbor given the same number of samples. **(c)** Estimation is more accurate if the training samples are more concentrated near the origin (template). **(d)** Performance drops when the test image is significantly more distorted than all the training images (The black dotted line shows the average magnitude of distortions $\|\mathbf{p}^{(i)}\|$ in the training images). **(e)** Using K_{NN}-nearest-neighbor with weighted voting reduces the number of training samples further.

our approach. Algorithm 2 is applied to each test image to obtain the relative (squared) error $e = \|\mathbf{p}_{\text{true}} - \tilde{\mathbf{p}}\|^2/\|\mathbf{p}_{\text{true}}\|^2$.

In the following, we discuss how to generate the warping bases and training samples.

Generation of PCA bases. The Gaussian Processes used to generate deformation field has zero mean and covariance function $k(\mathbf{x}_1, \mathbf{x}_2) = \exp\left(-\|\mathbf{x}_1 - \mathbf{x}_2\|^2/2\sigma^2\right)$. $\mathbf{x}_1$ and $\mathbf{x}_2$ are locations of pixels and σ is a hyper-parameter that keeps the deformation smooth.

From the generated deformation field, we apply PCA and pick the first 20 eigenvectors as the deformation bases. The standard deviations of the 1-st and 20-th principle components are $s_1 = 11.63$ and $s_{20} = 7.95$ respectively. This shows that the energy is evenly distributed among 20 dimensions, and there is no degenerated dimension. We use the standard deviation in generation of training samples.

Generation of training samples. We follow Eqn. (4.1) in generating training images. Eqn. (4.1) says once the training images are distributed, the distance between a randomly picked image at radius r in the parameter space and its nearby training image should be proportional to r. Thus the density $m(r)$ of training samples, as a function of r, is proportional to $1/r^d$, where d is the dimension of the parameter space.

$m(r)$ only characterizes the distribution along the radial axis. The assumption (Eqn. (4.1)) is in a spherically symmetric form and thus we set the angular distribution of training samples to be spherically symmetric. Thus, the radial density $m_l(r)$ (the density function after marginalizing

out all the angular components) is:

$$m_l(r) \propto m(r)\frac{\mathrm{dVol}_d(r)}{\mathrm{d}r} \propto \frac{1}{r} \tag{4.15}$$

where $\mathrm{Vol}_d(r)$ is the volume of d-dimensional sphere $\|\mathbf{p}\| \leq r$. As a sanity check, if Algorithm 2 returns the parameter with accuracy $1/\epsilon$, then along the radial axis, the training samples must be distributed along the interval $[\epsilon, r_0]$. By integrating $m_l(r)$ on this interval, we obtain:

$$\int_{\epsilon}^{r_0} m_l(r)\mathrm{d}r \propto \log r_0 - \log \epsilon = \log r_0/\epsilon \tag{4.16}$$

which is of the same order as Eqn. (4.13). Finally, Fig. 4.6 shows the distribution $m_l(r)$.

From Eqn. (4.15) we thus obtain an algorithm for sampling training distributions. There are two practical issues. Firstly, in order to show how the shape of training distribution affects the performance, instead of directly sampling from the distribution $m_l(r)$ (Fig. 4.6), we first sample r from a uniform distribution and exponentiate r by the *shape parameter* δ. For $\delta > 1$, this will also yield a distribution peaked around the origin, and in particular when $\delta \to +\infty$ it will give exactly the $1/r$ fall-off. Secondly, instead of using a uniform r_0 for all PCA coefficients, using the standard deviation of each PCA basis will increase the sampling efficiency.

Algorithm 3 Sampling training images

1: **INPUT** The required accuracy ϵ, the standard derivations $S = \mathrm{diag}(s_1, s_2, \ldots, s_d)$ of each PCA directions, the shape parameter δ and the number N of training samples.
2: **for** $n = 1 : N$ **do**
3: Draw sample r from a uniform distribution on $[0, 1]$ and exponentiate r by the shape parameter $\delta > 1$. A large δ yields peaked distribution around the origin.
4: Uniformly sampling the angular coordinates by drawing $\mathbf{v}$ from multivariate normal distribution $\mathbf{v} \sim \mathcal{N}(0, \mathbf{I})$ and normalize $\mathbf{v}$ so that $\|\mathbf{v}\| = 1$.
5: The n-th training sample $\mathbf{p}^{(n)} = rS\mathbf{v}$.
6: **end for**

Note that sampling using Algorithm 3 will yield the distribution that matches the *outwards decaying shape* as indicated by Eqn. (4.1). However, a fairly large number of training samples have to be drawn to achieve the *density* requirement of Eqn. (4.1), i.e. $\beta < 1$. The actual number of training samples depends on the complexity of manifold $\mathcal{I}$, the ratio of L_2/L_1 and how effective the nearest-neighbor matching is. In this experiment, we use $N = 1000$ if not explicitly mentioned and the algorithm works well.

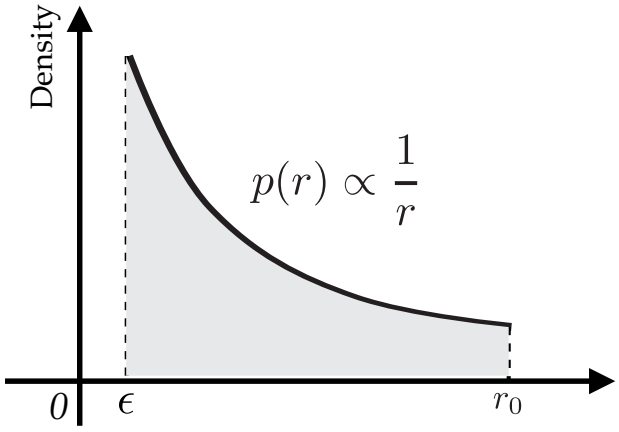


Figure 4.6: The radial density distribution $m_l(r)$ of training samples. Sampling from $m_l(r)$ (Algorithm 3) will yield the distribution that has the same *shape* as Eqn. (4.1) (yet β could be larger than 1). On the other hand, β, or the *density* of the distribution, is determined by the number of samples drawn.

Fig. 4.5(a) shows the successful convergence of our algorithm averaged over all the test images. Fig. 4.7 shows example images warped with different magnitudes of distortion and the computed rectified images. Particularly, notice the significant improvement in the most distorted example. Fig. 4.8 illustrates an image distorted by a 60 degree rotation. Even if a coarse-to-fine strategy is used, gradient-descent methods like Lucas-Kanade can get stuck in a local minimum due to the seemingly large displacement in the rotation angle. However, our algorithm converges successfully to the correct parameters in just 3 to 4 iterations.

4.5.2 Factors that affect the algorithm

There are four major factors that affect the performance of the algorithm, including **(a)** the number N of training samples, **(b)** the number K_{NN} of nearest-neighbors involved in prediction, **(c)** the shape parameter δ of the distribution of training images, and **(d)** the magnitude of distortion $\|\mathbf{p}_{\mathrm{true}}\|$ of the test images.

We set the default values of the four factors to be $N = 1000$, $K_{\mathrm{NN}} = 10$, $\delta = 2$ and $\|\mathbf{p}_{\mathrm{true}}\| = 30$. Fig. 4.5(b)-(e) shows performance variations when perturbing one factor and keeping the others constant. Fig. 4.5(b) shows better performance is obtained with more training images. Although nearest-neighbor behaves similarly, its performance is much poorer for the same number of samples. Fig. 4.5(c) shows that a high accuracy is obtained if training samples are concentrated around the origin (larger δ) given the test image is within their range, as supported by the theoretical analysis. Conversely, the performance drops gradually if a test image is far away from the training set (Fig. 4.5(d)). Finally, Fig. 4.5(e) shows that given the same set

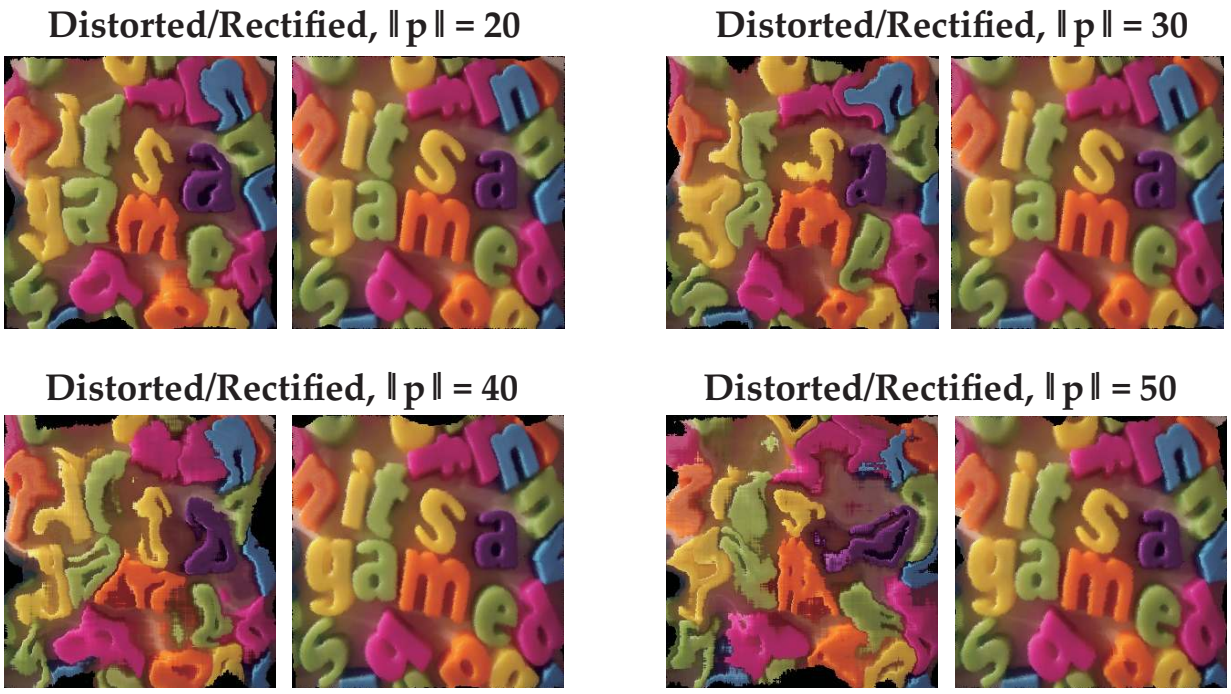


Figure 4.7: Sample images distorted to various degrees and the recovered rectified images. The template is shown in Fig. 4.4

of training samples, performance is better for K_{NN} nearest-neighbor with large K_{NN}. In other words, for the same performance, the parameter prediction using multiple neighbors requires fewer samples.

Verifying $\gamma < 1$ in Theorem 4.2.1. Fig. 4.9(a) shows how the distribution of relative prediction errors on the test images changes over iterations. The relative prediction error is defined as $\gamma_k \equiv \|\mathbf{p}^k_{\mathrm{true}} - \tilde{\mathbf{p}}^k\|/\|\mathbf{p}^k_{\mathrm{true}}\|$, which corresponds to γ in our theoretical analysis in Theorem 4.2.1. For 99.2% of the simulated distortions, the number of samples (1000) we used are sufficient and $\gamma_k < 1$, indicating the algorithm's convergence. For the remaining 0.8%, the simulated distortions were too large and without sufficient training samples, hence $\gamma_k \geq 1$. The distributions of γ_k show that the rate of convergence slows down with increasing iterations. This is because more samples would be required around the origin to achieve a higher accuracy.

Performance under severe image resampling artifacts. Recall that resampling artifacts are not considered in our theoretical analysis. For large distortions where resampling artifacts can be overwhelming, our algorithm may not have the desired behavior. Interestingly, for many such cases, the observed difference between the rectified image and the template has the same shape as the actual distance between the true parameters and the estimated parameters (see Fig. 4.9(b)).

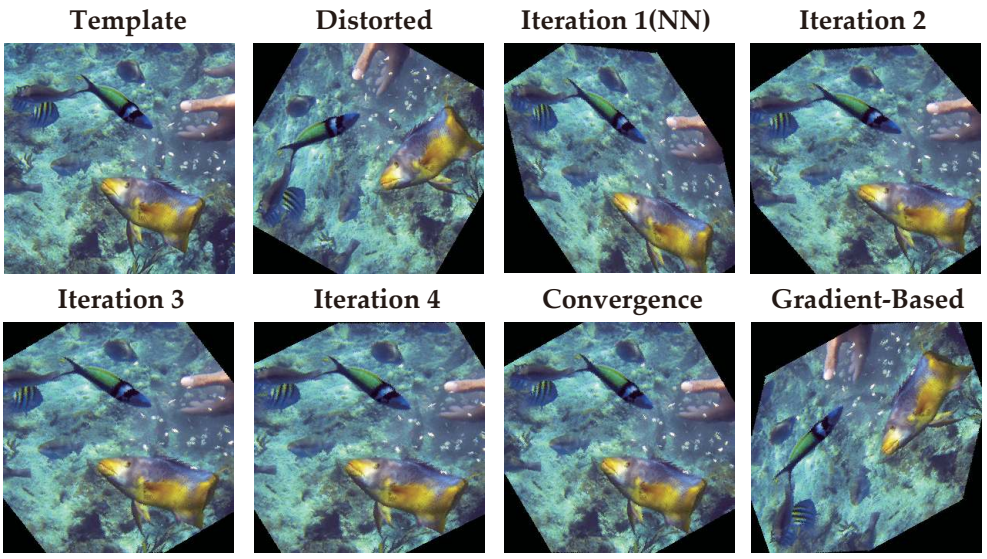


Figure 4.8: Successful convergence of our algorithm for affine transformed image, given there is at least one training sample reaching that area. In contrast, gradient-descent methods (like Lucas-Kanade [6]) get stuck in local minima even with a coarse-to-fine strategy.

Hence, we conjecture that the solution that produces minimum error in the image metric among many iterations will be a reasonable one, which is used as the stopping criterion in the real experiments.

4.5.3 Performance in the presence of noise and occlusion

We also check the usability of our method in the presence of noise and occlusion. In this experiment, we use the same 100 images as in Section 4.5.1. For each image, 1000 samples are generated as training and 10 samples as testing. Each test image is contaminated with salt & pepper noise or rectangle-shaped occlusion before our algorithm is applied. To generate the salt & pepper noise, we randomly choose a portion ρ_{P} of pixels in the test image and set their values randomly (uniformly distributed in $[0, 1]$). In the case of rectangle-shaped occlusion, we choose a random position of a rectangle whose area is a portion ρ_{R} of the entire image, and fill in this rectangle with random noise that is uniformly distributed in $[0, 1]$. We use two pixel-wise image metrics, l_1 and l_2-norm on grayscale images, for nearest-neighbor.

Table 4.1 and Table 4.2 show our method is relatively robust to noise and occlusion in both cases. When the noise level is 10%-30%, our method still gives a reasonable estimation of distortion, with slightly increased squared prediction errors in the parameter space. Especially, l_1 metric performs better than l_2 metric in the rectangle-shaped occlusion case for occlusion rate up to 40%. Our method contrasts with many gradient-based approaches, in which a robust

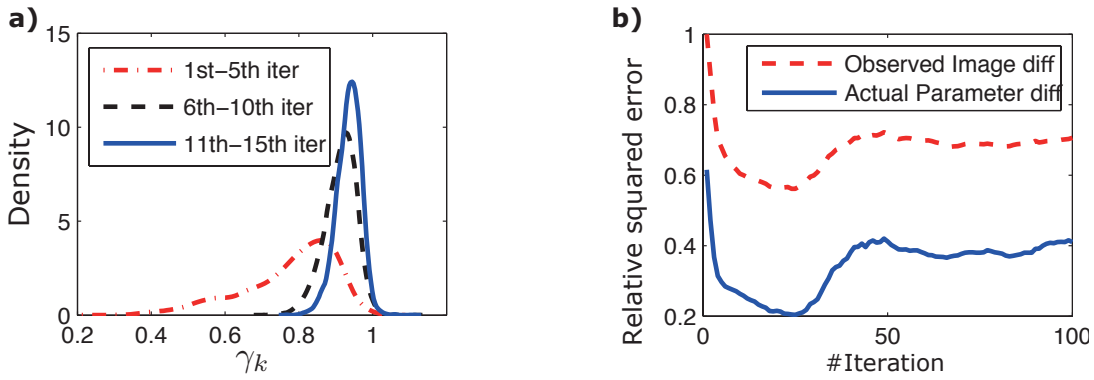


Figure 4.9: **(a)** The empirical distribution of relative prediction error γ_k on test images in different iterations of the algorithm. 99.2% of the γ_k is small than 1, justifying $\gamma < 1$ in Theorem 4.2.1; others are due to insufficient samples. **(b)** The U-turn behavior in large distortion ($\|\mathbf{p}_{\text{true}}\| = 50$), when the resampling artifacts are severe.

Mild distortion ($\|\mathbf{p}\| = 20$)						
ρ_{P}	No occ	10%	20%	30%	40%	50%
l_2-norm	0.0646	0.0644	0.0668	0.0729	0.0863	0.1196
l_1-norm	0.0383	0.0419	0.0476	0.0599	0.0973	0.2440
Moderate distortion ($\|\mathbf{p}\| = 30$)						
ρ_{P}	No occ	10%	20%	30%	40%	50%
l_2-norm	0.0587	0.0607	0.0651	0.0751	0.0939	0.1427
l_1-norm	0.0363	0.0411	0.0481	0.0649	0.1195	0.2987
Large distortion ($\|\mathbf{p}\| = 40$)						
ρ_{P}	No occ	10%	20%	30%	40%	50%
l_2-norm	0.0595	0.0630	0.0703	0.0853	0.1164	0.1981
l_1-norm	0.0469	0.0508	0.0630	0.1009	0.2002	0.4207

Table 4.1: Relative squared errors of the estimated distortion of test images with salt & pepper noise. Note ρ_{P} is the percentage of contaminated pixels in the test image.

Mild distortion ($\|\mathbf{p}\| = 20$)						
ρ_{R}	No occ	10%	20%	30%	40%	50%
l_2-norm	0.0646	0.0686	0.0796	0.1202	0.2488	0.5146
l_1-norm	0.0383	0.0417	0.0486	0.0544	0.0858	0.6513
Moderate distortion ($\|\mathbf{p}\| = 30$)						
ρ_{R}	No occ	10%	20%	30%	40%	50%
l_2-norm	0.0587	0.0656	0.0825	0.1369	0.2292	0.4659
l_1-norm	0.0363	0.0431	0.0510	0.0772	0.1253	0.3055
Large distortion ($\|\mathbf{p}\| = 40$)						
ρ_{R}	No occ	10%	20%	30%	40%	50%
l_2-norm	0.0595	0.0729	0.1021	0.1624	0.3028	0.5437
l_1-norm	0.0469	0.0563	0.0821	0.1606	0.2937	1.2850

Table 4.2: Relative squared errors of the estimated distortion with rectangle occluded test images. Note ρ_{R} is the percentage of occluded pixels in the test image.

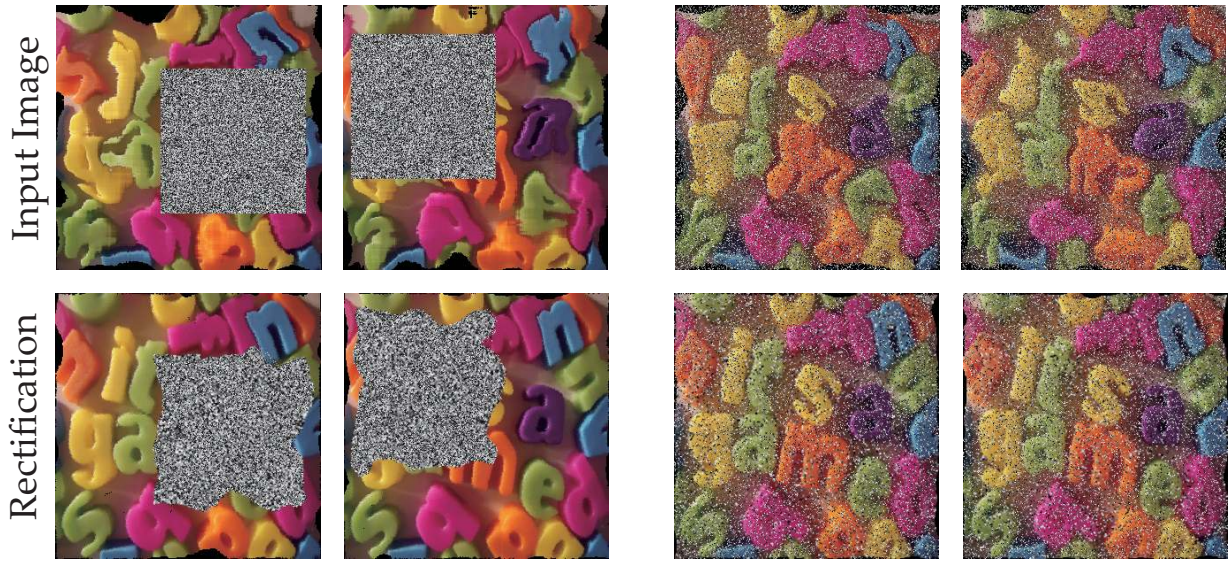


Figure 4.10: Distorted test images with noise/occlusion and their rectifications. **(a)** Distorted images with rectangle-shaped occlusion. **(b)** Distorted images with salt & pepper noise. Despite a large portion of the distorted image is contaminated, our algorithm still obtains a reasonable estimation of the distortion parameter and rectifies the image correctly. In all the results, we use the setting $\rho = 30\%$ and $\|\mathbf{p}\| = 30$. Note the algorithm is run on grayscale images and color is used here merely for illustration. (Best viewed in color)

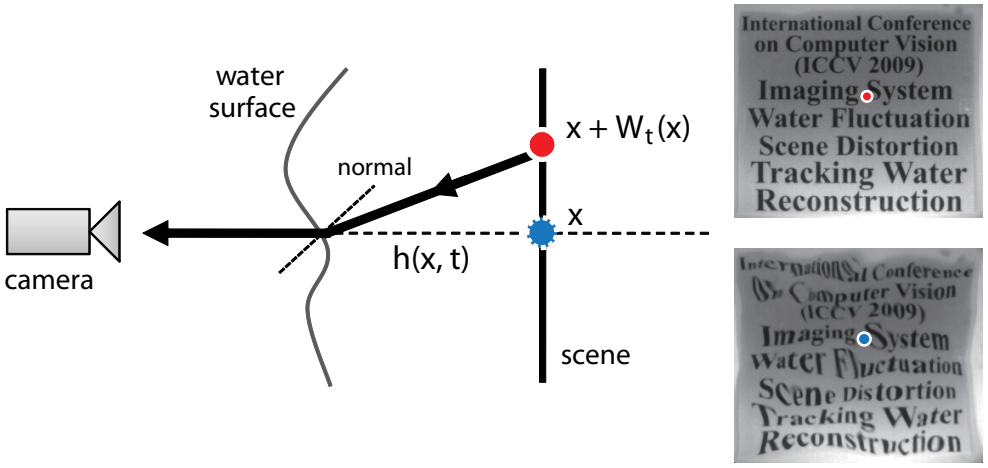


Figure 4.11: Image formation in the presence of water distortion. The scene pixel at $\mathbf{x} + W_t(\mathbf{x})$ is perceived at location $\mathbf{x}$ in the distorted image.

distance measure or a reweighting scheme has to be involved, and the initial parameters have to be carefully chosen.

4.6 Application I: Imaging through Water

The shapes of many deformable and time-varying interfaces between two media with different refraction indices, such as water surface, are very hard to measure directly. By perceiving the distortion of underwater scene, human vision can sense the fluctuation of the water surface qualitatively. In the following, we show that using Algorithm 2, we can estimate quantitatively the shape of the water surface, given both the appearance of the underwater scene when the water surface is still and a distorted image due to water fluctuation. This approach also works for a distorted video sequence by applying the same algorithm per frame. As a result, the shape of the water surface can be estimated over time.

4.6.1 Distortion Bases

Since the water distortion is caused by the bending normals of the water surface, its distortion bases can be obtained by physical simulation of water. According to Snell's law (Fig. 4.11), under first-order approximation, we can relate the distortion $W_t(\mathbf{x})$ to the height $h(\mathbf{x}, t)$ of the water surface at each time t:

$$W_t(\mathbf{x}) = \eta \nabla h(\mathbf{x}, t) \tag{4.17}$$

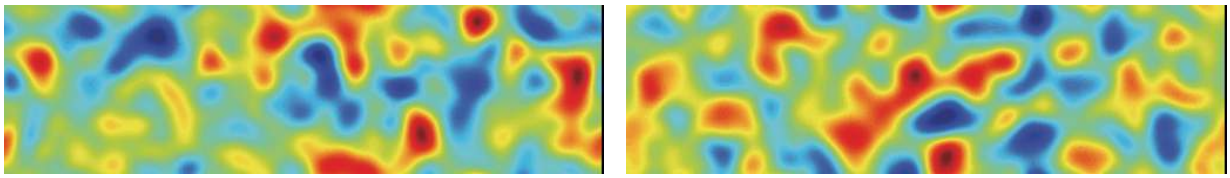


Figure 4.12: Two samples of 2-D Gaussian processes used as the initial conditions of the wave simulator (Eqn. (4.19)).

where η is a constant related to water height h_0 when the water surface is still, and relative refraction index between air and water. When the maximum surface fluctuation

$$\max_{\mathbf{x},t} |h(\mathbf{x},t) - h_0| \tag{4.18}$$

is small compared to h_0, the water surface is governed by the following wave equation:

$$\frac{\partial^2 h(\mathbf{x},t)}{\partial t^2} = c^2 \nabla^2 h(\mathbf{x},t) \tag{4.19}$$

where $c = \sqrt{g h_0}$ is the velocity of wave (g is the gravity).

To simulate the wave equation, we use forward Euler method with a periodic boundary condition. This strategy is easy to implement and stable for small time step Δt:

$$h(\mathbf{x}, t+\Delta t) = 2h(\mathbf{x},t) - h(\mathbf{x}, t-\Delta t) + c^2 \nabla^2 h(\mathbf{x},t)(\Delta t)^2 \tag{4.20}$$

where $\nabla^2 h(\mathbf{x},t)$ is the Laplacian operator on the water height image at time t. The initial conditions $h(\mathbf{x},0)$ and $h(\mathbf{x},\Delta t)$ are chosen to be a spatially correlated Gaussian Processes in a 2-D grid, as illustrated in Fig. 4.12. More specifically, $h(\mathbf{x},0)$ and $h(\mathbf{x},\Delta t)$ are sampled from a multivariate Gaussian distribution $N(h_0\mathbf{1}, \Sigma)$ with each entry of the covariance $\Sigma_{\mathbf{x},\mathbf{x}'}$ inversely proportional to the spatial distance between $\mathbf{x}$ and $\mathbf{x}'$:

$$\Sigma_{\mathbf{x},\mathbf{x}'} = \exp\left(-\frac{\|\mathbf{x}-\mathbf{x}'\|^2}{2\sigma^2_{\text{synthesis}}}\right) \tag{4.21}$$

Note both the mean and variance of the Gaussian distribution are independent of the absolute coordinates of spatial locations. Thus the resulting initial condition is spatially stationary. $\sigma_{\text{synthesis}}$ is set by visually comparing the appearance of a known underwater planar scene at the bottom of the water tank with that from simulations. Importantly, $\sigma_{\text{synthesis}}$ is independent of the underlying scene. In the simulation, we set $c = 0.8$ pixel/frame and $\sigma_{\text{synthesis}} = 10$ pixels.

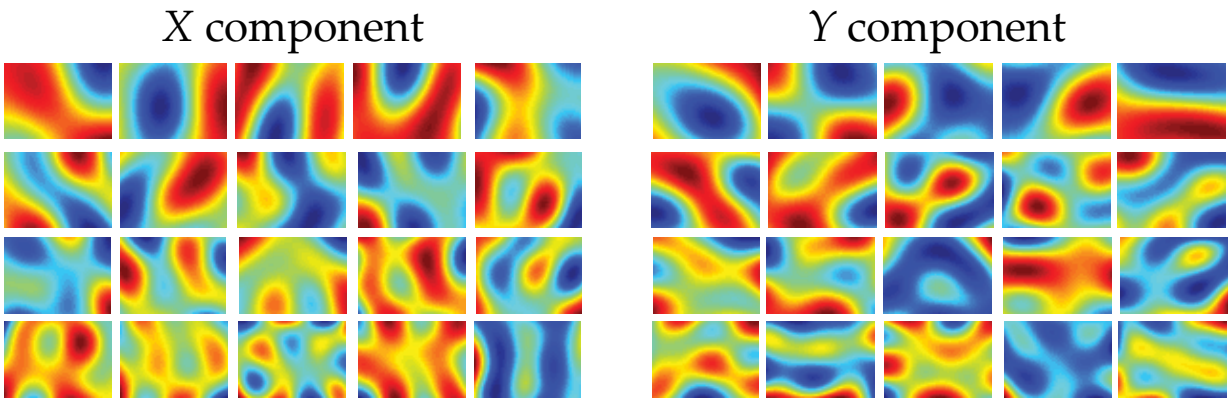


Figure 4.13: The water bases $B(\mathbf{x}) = [\mathbf{b}_1(\mathbf{x}), \mathbf{b}_2(\mathbf{x}), \ldots, \mathbf{b}_{20}(\mathbf{x})]$. For both x and y components, the bases are sorted by their eigenvalues in a descending order, from left to right and from top to bottom.

The simulator gives the time-evolving shape of the water surface. Since the initial condition is spatially stationary, and the wave equation is a time-invariant partial differential equation, we conclude that the evolving water surface is both temporally and spatially stationary. Thus, it suffices to capture the statistical properties on local patches. Based on this insight, we randomly sample space-time coordinates $(\mathbf{x}, t)$ and extract spatial patches (57×40) from W_t centered at $\mathbf{x}$. Then PCA is applied to these sampled patches to obtain the first 20 orthogonal principle modes $B(\mathbf{x}) = [\mathbf{b}_1(\mathbf{x}), \mathbf{b}_2(\mathbf{x}), \ldots, \mathbf{b}_{20}(\mathbf{x})]$ of water distortion, which we call *water bases* as shown in Fig. 4.13. The standard deviations of the 1-st and 20-th principle components are 610.08 and 42.82 respectively. By construction, the bases are translation invariant.

4.6.2 Experimental Setup

The water experiment consists of video camera observing vertically downward a 0.5m deep semi-transparent water tank with a planar scene at the bottom. The tank is illuminated from the side to avoid any surface reflections that are not modeled. The water surface is manually disturbed using a plastic ruler. The planar scene includes fonts of various sizes and natural textured underwater scene. The average dimension of distorted video sequences is around 350×250 with 500 frames. The variations of the dimension are due to a manual preprocessing step to trim the image boundaries corresponding to the water tank.

We use the image taken under flat water surface as the template. Since the water distortion is local, we partition the image into overlapping patches and apply Algorithm 2 with the water bases (Fig. 4.13) on each patch to obtain a local deformation field. The image distance is computed using l_1 metric in grayscale after normalizing the pixel intensity into $[0, 1]$. 10000 training

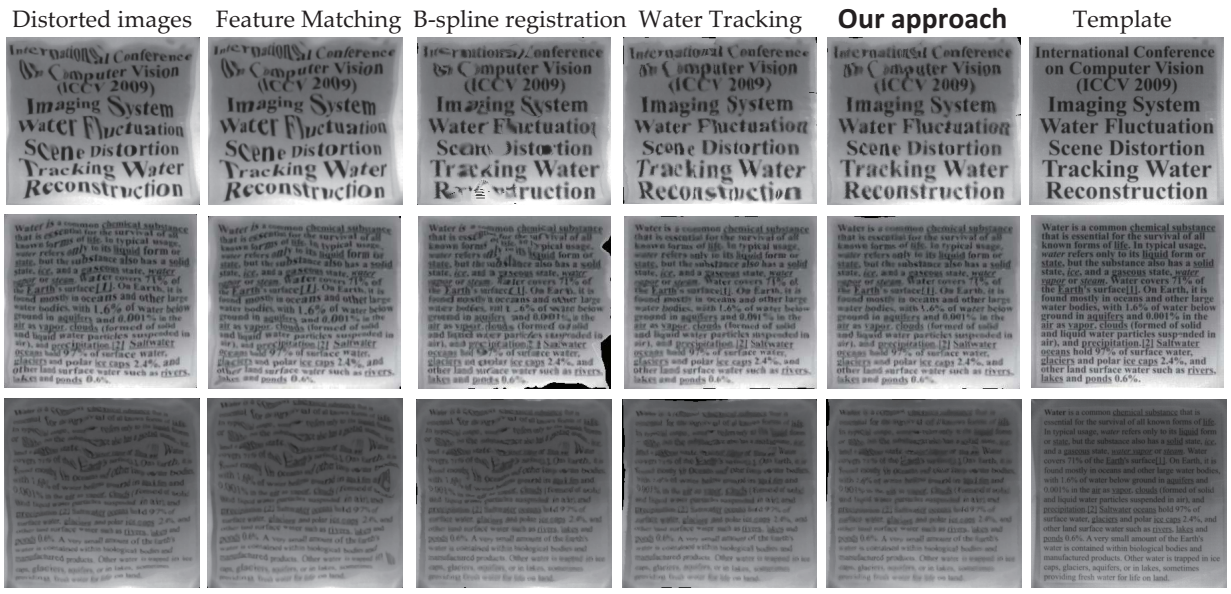


Figure 4.14: Rectification of water distortion on text images of different font sizes (from the top row to the bottom row: MiddleFonts, SmallFonts and TinyFonts). Our approach outperforms HOG (Histogram of Gradient) feature matching, B-spline nonrigid registration [69] and yields slightly better results with water tracking [98]. However, water tracking relies on the entire video frames, while ours only needs two images.

samples are synthesized from the template using the water bases, densely distributed around the original but sparsely elsewhere, as described in Section 4.5.1. For each distorted patch in the video sequence, 15 iterations are performed to obtain the parameter estimation on the water bases. Then these local deformation fields are stitched together, resulting in a global deformation field. At the overlapping regions between patches, we average the local deformation fields given by neighboring patches to obtain a smooth transition.

4.6.3 Results

Rectification of distorted images. We compared our algorithm to several previous representative techniques: free-form non-rigid image registration using B-splines [69], our previous work of water tracking [98] and a baseline approach in which we compute and match HOG (Histogram of Gradient) descriptors and interpolate the sparse correspondence using thin-plate interpolation to create a dense deformation field. We also compare with the classic Lucas-Kanade method with the same set of water bases plus a coarse-to-fine strategy, as shown quantitatively in Section 4.6.4.

Fig. 4.14 shows the rectified images for a scene with text, and Fig. 4.19, 4.20 shows the results for a scene with colored textures. All the datasets, including three scenes with text (tiny-

Figure 4.15: Tracking a video sequence using estimated deformation fields. Although the underlying fish images are non-rigidly distorted, our method can still track it without drifting, using only grayscale images (We show color images for better illustration). Note the contour of the object in the first frame is manually labeled. See our website for the complete video sequence.

Font, middleFont and smallFont) and scenes with textures, can be downloaded in our website. Since only sparse correspondences between two images are used, feature matching gives an inaccurate interpolated deformation field and fails to align details well. Nonrigid B-spline image registration [69] works better but fails occasionally on some image regions due to local minima. Our previous method, water tracking [98] produces better results than feature matching and B-spline registration. Yet it requires a short video sequence (61 frames) to rectify a single frame. In contrast, our method yields the best rectification results given only the template and one distorted image at a time.

Video tracking. Using the estimated distortion, one can find the corresponding points of an object's contour at each video frame, which gives the tracking result as shown in Fig. 4.15. We can see that although the shape of the fish undergoes large nonrigid distortions, our method still succeeded in tracking its contour reliably (note the first contour is manually labeled).

Water surface reconstruction. According to Eqn. (4.17), the deformation fields are proportional to the gradient of the water height at any time. Hence, one can recover the height of the water surface at each time using Frankot-Chellappa integration [26] on dense deformation fields of x and y directions. Some sample reconstructions are shown in Fig. 4.16.

Please check more video results on our website.

4.6.4 Quantitative Evaluation

In addition to visual comparisons, we also do quantitative comparisons to further verify our approach.

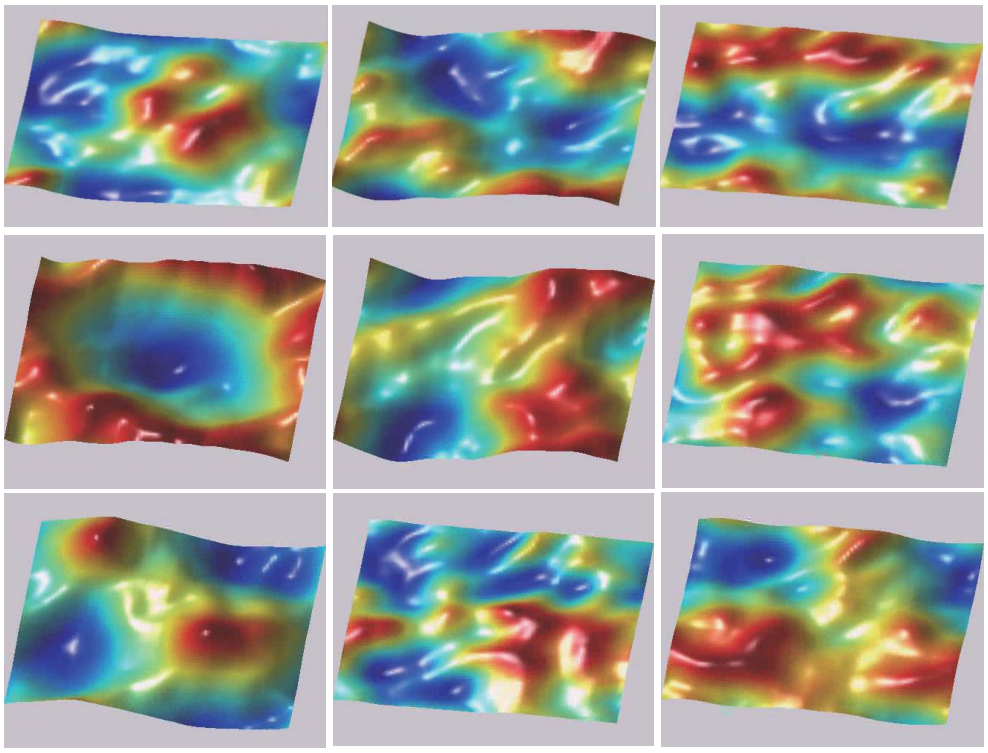


Figure 4.16: Reconstructed water surfaces (dataset: SmallFonts) by spatially integrating the water distortion (Best viewed in color).

Reprojection error on images

Without groundtruth deformation fields, a convenient evaluation is to check whether the rectified frames coincide well with the template, which is the image reprojection error. We compare our method with B-spline registration [69] and Lucas-Kanade registration using the same water bases (Fig. 4.13) and a coarse-to-fine strategy to avoid possible local minima. To measure the distance between a rectified image I and the template I_0, we compute the root-mean-square reprojection error $RMS_{\text{intensity}}$ as follows:

$$RMS_{\text{intensity}} = \sqrt{\frac{1}{n}\sum_{\mathbf{x}}(I(\mathbf{x}) - T(\mathbf{x}))^2} \tag{4.22}$$

where n is the number of pixels in each image. Note the image intensity is normalized into $[0, 1]$ before different algorithms are formally applied. For a video sequence, we compute RMS for each frame and take the mean value over time. Table 4.3 shows the result. We can see even with the same bases, Lucas-Kanade still gets trapped into the local minima and fails to give a low reprojection error. B-spline works better yet our method performs the best.

Dataset	Distorted video	Lucas-Kanade	B-spline [69]	Our method
TinyFonts	0.0720	0.0618	0.0553	**0.0444**
SmallFonts	0.1029	0.0624	0.0512	**0.0461**
MiddleFonts	0.1551	0.1092	0.0640	**0.0597**
Fish	0.0995	0.0831	0.0584	**0.0527**

Table 4.3: Comparison of the image reprojection error on different methods. All the errors are computed using RMS (See Eqn. (4.22)) and the mean RMS over the entire video sequence (500 frames) is shown in the table. Note the pixel intensity is normalized into $[0, 1]$ before different algorithms are applied. Thus the maximal possible reprojection error is 1 (black versus white images).

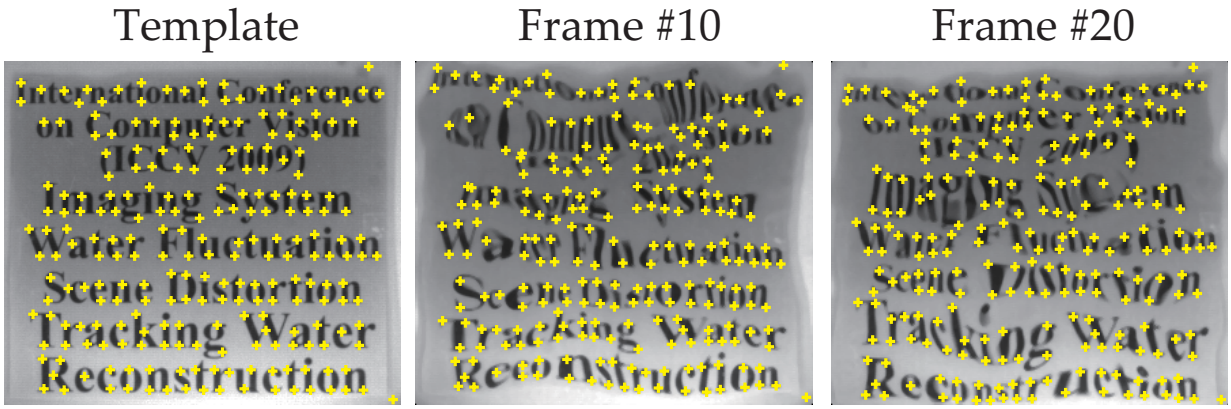


Figure 4.17: Samples of landmark-labeled frames in dataset MiddleFonts. Note the video frames and the template are 253 by 293. The first 30 frames are manually labeled, each with 232 landmarks.

Reprojection error on landmarks

The image reprojection error is not a perfect performance measure; a distortion estimation algorithm may result in lower errors by arbitrarily rearranging the pixels without considering the spatial smoothness constraints. To further verify our method, we manually label $m = 232$ landmarks on the first 30 frames of one of the underwater dataset, MiddleFonts (See Fig. 4.17 for sample labels), and compute root-mean-square error RMS_{spatial} between the landmark positions $\{\mathbf{x}_i^{\text{t}}\}$ transformed from the template to the distorted frame using the estimated deformation field, and the landmark positions $\{\mathbf{x}_i^{\text{d}}\}$ that are labeled on the distorted frame:

$$RMS_{\text{spatial}} = \sqrt{\frac{1}{m}\sum_{i=1}^{m} \|\mathbf{x}_i^{\text{t}} - \mathbf{x}_i^{\text{d}}\|_2^2} \tag{4.23}$$

	Distorted video	Lucas-Kanade	Feature matching	B-spline [69]	Our method
mean RMS	6.3404	5.2040	3.9282	3.8212	**2.5142**

Table 4.4: Comparison of the landmarks reprojection error on different methods. All the errors are computed using Eqn. (4.23) and in the table the mean error over the 30 labeled video frames of the Middle-Font dataset is shown. See Section 4.6.4 for detailed descriptions of each listed method.

Similarly, we compute mean RMS over 30 labeled distorted frames. Table 4.4 shows the results. We can see our method gives the smallest errors (measured in pixel), while other generative approaches, such as Lucas-Kanade (with the same set of bases) and B-spline, gives at least 60% higher errors. Since the landmark correspondence is sparse, we also test the performance of feature matching using HOG descriptor. To minimize the matching ambiguity and using the prior knowledge that the landmark positions are fluctuated around their positions in the template, we match each HOG descriptor located at $\mathbf{x}$ in the template with all the *densely* extracted descriptors located in the vicinity of 11 pixels in the distorted frame, and pick the best one as the matching result. This approach yields better results than Lucas-Kanade and comparable to B-spline, yet is still not as good as our approach. Finally, Fig. 4.18 gives a more detailed analysis of the error distribution of different methods.

4.7 Application II: Cloth Deformation

Another interesting application of Algorithm 2 is to estimate nonrigid cloth deformation. Given a video sequence with deforming cloth, the goal is to estimate a dense and time-varying deformation field between different frames, which can be used for video tracking and 3D reconstruction.

4.7.1 Global motion and local deformation

In general, since cloth deformation behaves more globally than water distortion, we use the following two-stage approach. In the first stage, we downsample the original video (720×480) by a factor of 2, apply local affine bases of size 200×200 and estimate its 6 parameters using our method. This gives a coarsely undistorted video sequence. In the second stage, we apply local random bases (100×100) with 40 dimensions to the undistorted sequence, and obtain the final distortion estimation by distortion composition. We build our own dataset acquired by manually perturbing a piece of silk cloth with repetitive heart patterns.

In addition, we apply our method on the dataset offered by the authors of [95] to obtain the dense deformation field, which is used to reconstruct the 3D shape of the cloth. For their datasets,

we use a slightly different approach. We begin by first using local trackers (mentioned below) to track reliable interest points over time and manually pick the correct trackers to obtain a *coarse* dense deformation field with thin-plate interpolation. Then local random bases are again applied on the coarsely rectified video sequences for refined estimation.

The local tracker. The local tracker we used is also based on Algorithm 2. Given an interest point in the template (usually is the first frame of the deforming cloth sequence), a local patch around it is cropped and 200 samples are generated using affine warps. During tracking, we initialize the position of the tracker as its position in the previous frame and extract the patch around it, on which Algorithm 2 applies to obtain the local deformation field that gives the position of the tracker in the current frame. With an illumination-invariant metric, this local tracker is robust to the shading effects in the cloth video sequence. As a result, many of the tracking trajectories are reliable and useful throughout the video sequence. By manually picking the good ones, a coarse yet representative deformation field can be built.

4.7.2 Results

Fig. 4.22 shows some sample frames of a rectified video sequence produced by our method on a piece of cloth with repetitive heart patterns. B-spline registration [69], as a generative approach, goes into local minima in multiple frames, while our approach does not. Please watch the entire video sequence on our website for a more thorough comparison. Fig. 4.21 shows the estimated deformation fields. The affine components are shown as the linear part of the deformation fields, while the nonrigid components are shown as the nonlinear part, as clearly illustrated in this figure.

Fig. 4.23 shows the established correspondence on the data-set from [95]. Our method captures the wavy structure on the cloth in the first dataset and the bending structure in the second dataset throughout the video sequence. The 3D reconstruction of the dataset can be found in [95].

4.8 Applications III: Air Turbulence

Using deformation estimation, we can recover the scene under turbulence from a set of image sequence. With short exposure, every image of the sequence is a noisy and distorted version of the scene. A long exposure reduces the two artifacts but introduces blurring. Therefore, how to obtain an image of the scene without all the artifacts is not a trivial problem. Following a similar setting, [114] combines a noisy and a blurry image of the same scene to obtain a better one. In our case, each frame of the video sequence is not only noisy but also misaligned.

As shown in Fig. 4.24 and Fig. 4.25, the sample frame from the sequence (the first column)

is noisy and distorted; while the mean image of the frames (the second column) is blurry. Note that the mean image could be regarded as an image captured with very long exposure. To obtain a better image of the scene, we first densely register the image sequence to the first frame with Data-Driven Descent to obtain frame-to-frame warps. Then we rectify each image of the sequence and take their mean. Due to some alignment error and blur occurred in low exposure frames, the resulting rectified mean is still blurry (but less than the original mean), and thus we deblur it with Richardson-Lucy (RL) method. The final result is shown in the last columns.

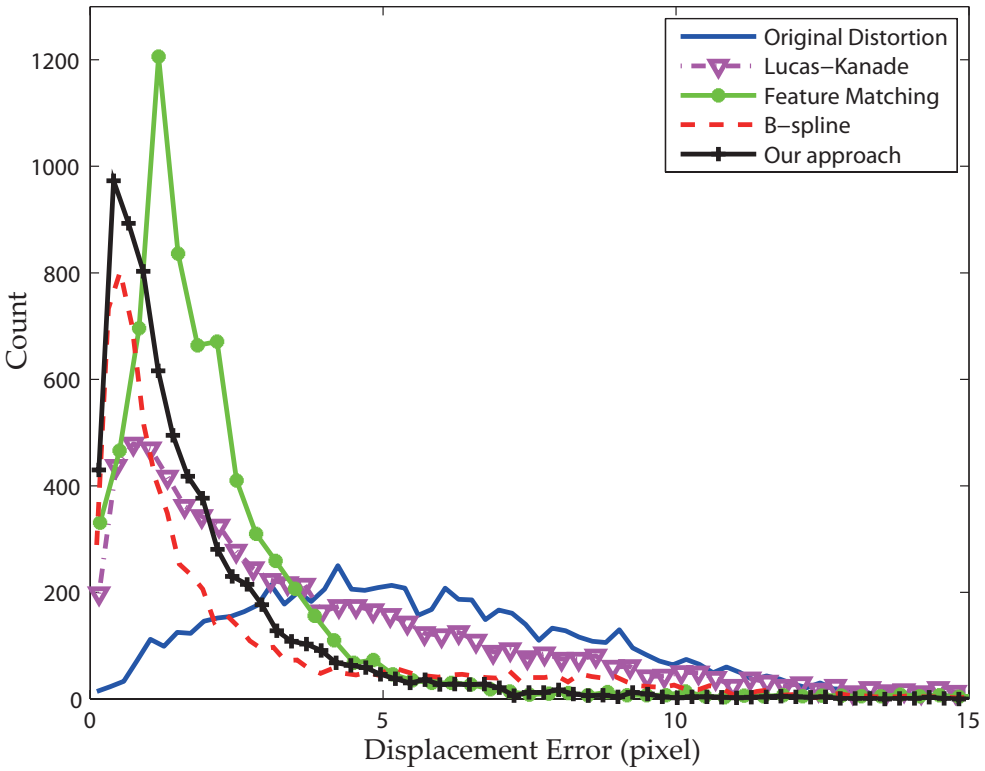


Figure 4.18: Histograms of landmark displacement errors using different methods over 30 labeled frames, each with 232 landmarks. The displacements in the distorted images (blue solid line) follow a flat and Gaussian-like distribution. All the methods aim to push the distribution towards the origin. The Lucas-Kanade method (magenta line with triangle) produces a error distribution with a heavy tail, indicating that it often converges to local optima and many landmarks fail to align well. Local dense feature matching (green line with circle) works better, but the local ambiguity of HOG features leads to inaccuracy in the alignment, as indicated by the sharp peak of the distribution located at a region of positive errors. B-spline registration [69] (dashed red line) works even better using a more powerful optimization technique (BFGS) but still not as good as our method (black line with cross) whose error distribution is more concentrated near the origin and with a thinner tail.

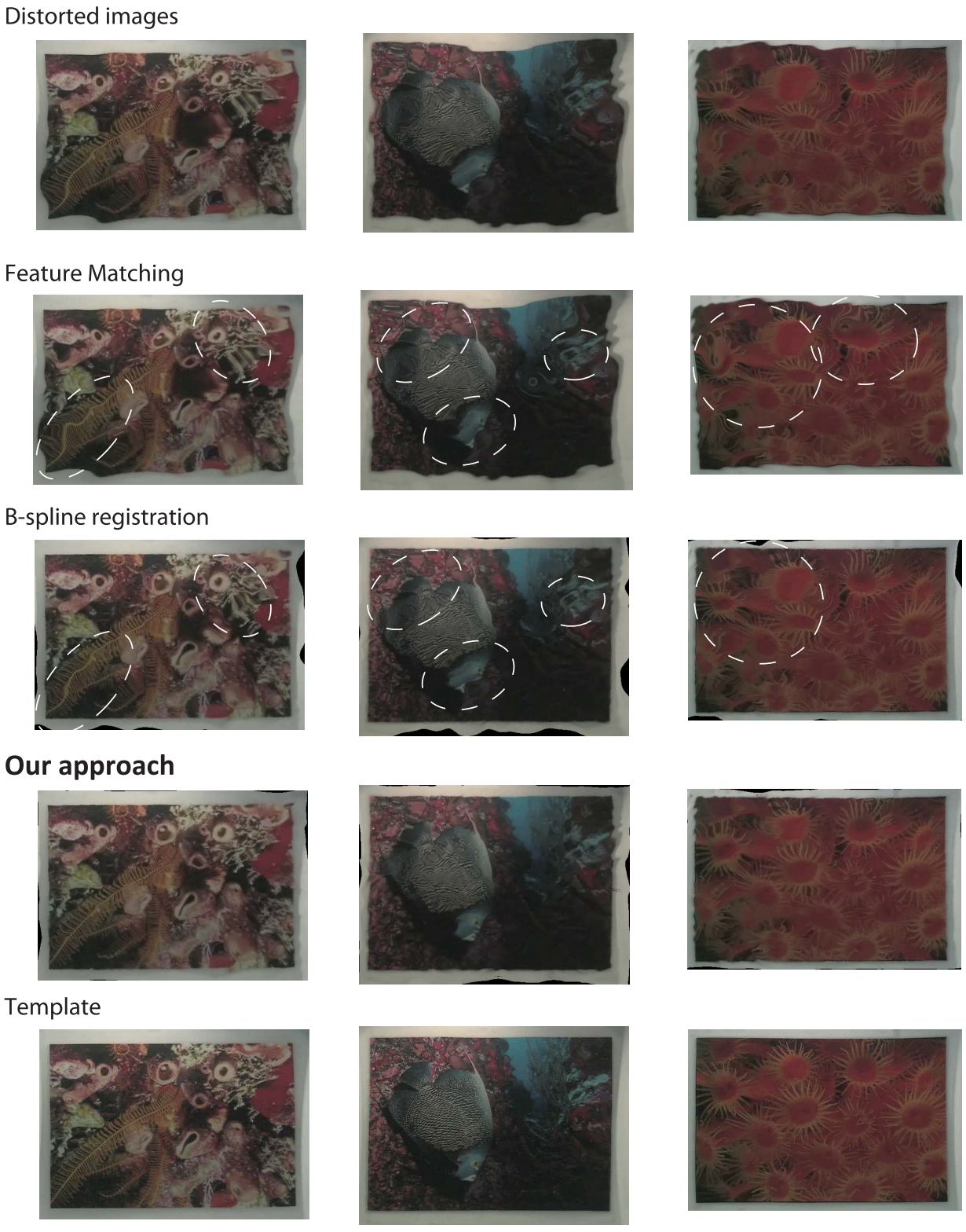


Figure 4.19: Rectification of water distortion on 3 different colored texture images. Our method yields the best rectification. Detailed comparison is shown in Fig. 4.20 (Best viewed in color).

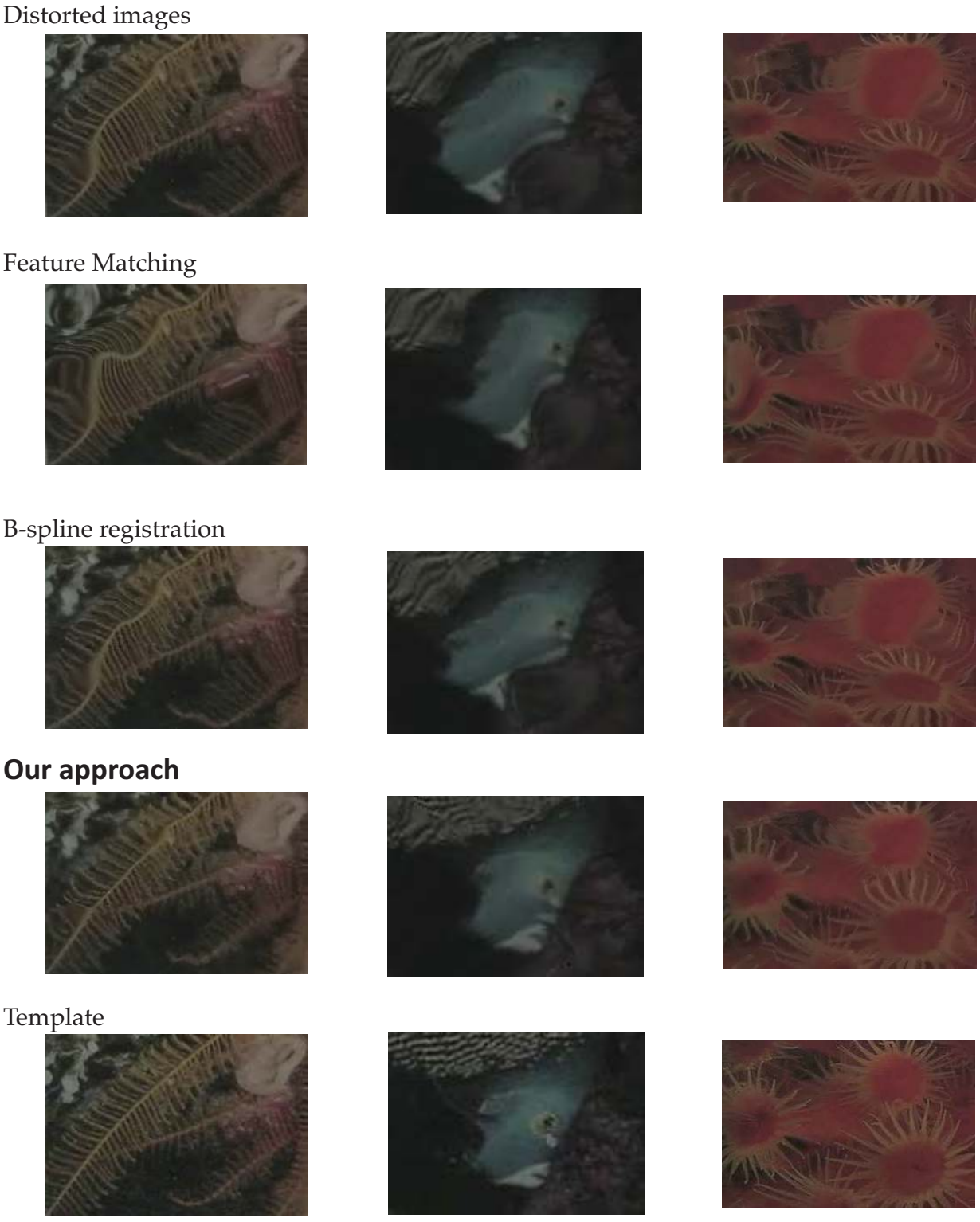


Figure 4.20: Detailed comparison between our approach and previous works [69]. (Best viewed in color)

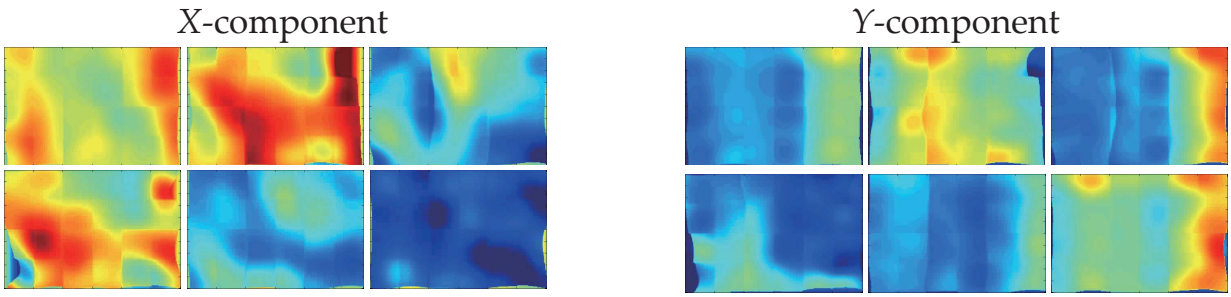


Figure 4.21: Estimated deformation fields for the cloth sequence with repetitive heart patterns. The first row shows the x-component while the second row shows the y-component. The linear part in distortion fields is the affine component, while the nonlinear part is the nonrigid component. (Best viewed in color)

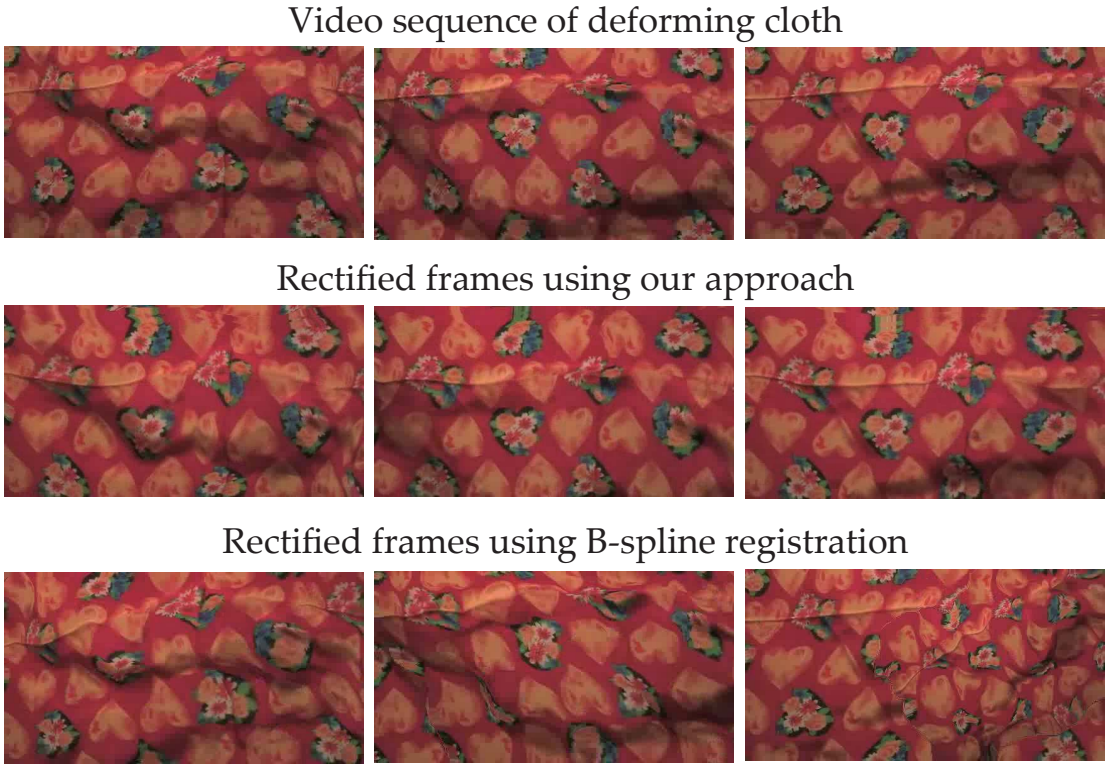


Figure 4.22: Rectification of cloth deformation using different methods. The first row shows the original video frames, the second row shows the rectified video frames by our approach, and the last row shows the rectification by B-spline registration [69]. As a generative approach, B-spline registration converges to local minima; while our approach gives good distortion estimation and rectifies the deformation correctly.

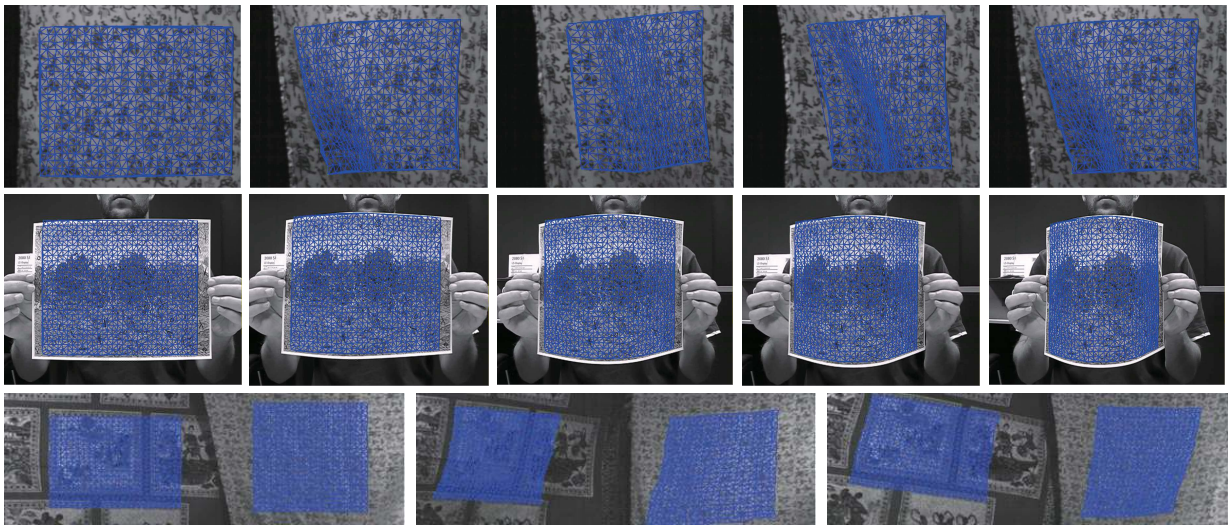
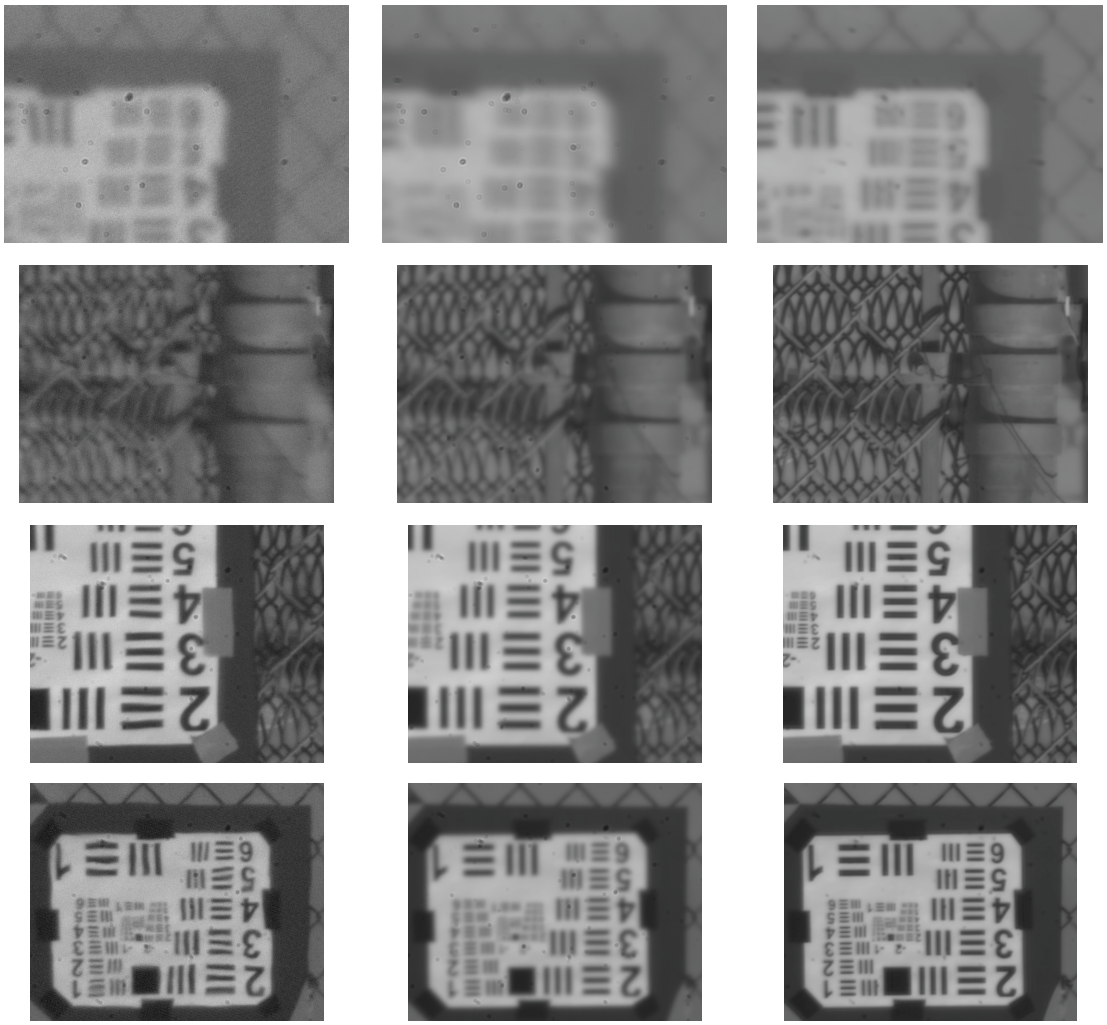
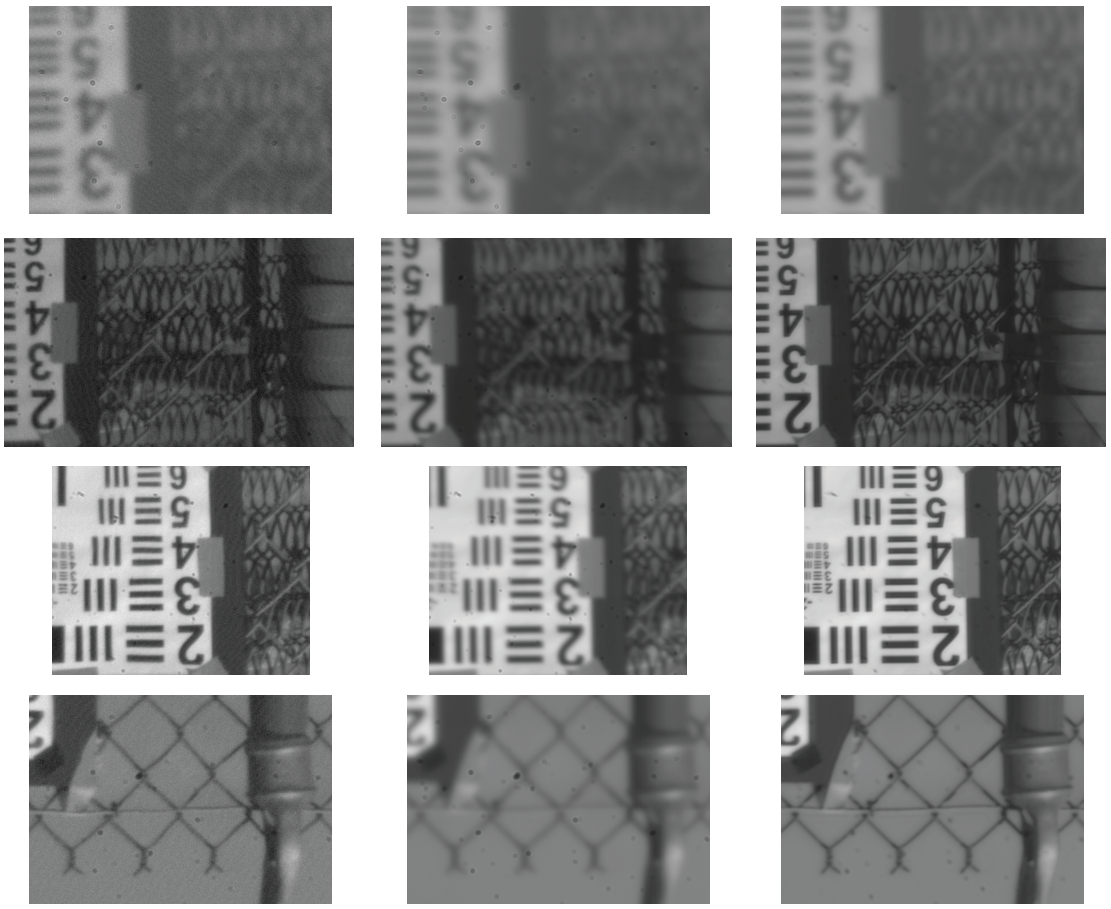


Figure 4.23: Estimated 2D mesh on the video sequence of deforming cloth using our approach. The dataset in the first and the last row come from [95], while the dataset in the middle row comes from [72]. (Best viewed in color)



(a) Sample Frame (b) Mean Image (c) Rectified using [1]

Figure 4.24: Scene Rectification. **(a)** Sample frame from a video. The image is sharp but noisy. **(b)** Mean image of the video. The image is noise-free but blurry. **(c)** Rectification using Data-Driven Descent.



(a) Sample Frame (b) Mean Image (c) Rectified using [1]

Figure 4.25: Scene Rectification. **(a)** Sample frame from a video. The image is sharp but noisy. **(b)** Mean image of the video. The image is noise-free but blurry. **(c)** Rectification using Data-Driven Descent.

Part II

Theory II: From Image to Hierarchy

Chapter 5

Patch-based Representations and The Construction of Hierarchy

In the previous chapters, the proposed Data-driven Descent has successfully reduced the sample complexity from $O(1/\epsilon^d)$ to $O(C^d \log 1/\epsilon)$ while maintaining the global optimality guarantees. To further reduce the sample complexity, in the experiments, the entire image of size m-by-n is partitioned into small image regions whose dimension is much smaller than the entire image. Such regions are often called patches.

Patches and patch-based representations are not new, and have been extensively used in computer vision due to many advantages over holistic approaches. For example, in image inpainting, missing regions can be filled in using identical patches elsewhere in the same image. In feature-based image matching, patches are used to extract features such as SIFT [53] for finding sparse correspondences between two or more images. In object recognition, patches are the building blocks for textons [87] and bags of words [77] due to its high repetitive nature and (limited) discriminative power.

From the point of view of this thesis, one major advantage of patch-based representation is its low dimensionality. Under the smoothness constraint of deformation field, the dimensionality (degrees of freedom) of an image region R is at most proportional to its area. Since the sample complexity grows exponentially with respect to the dimensionality, using a patch rather than the entire image could reduce the sample needed substantially. Indeed, in Chapter 4, Data-driven Descent is applied independently to the patches of water-distorted video frames.

This sounds very attractive. Then why not go to the other extreme, i.e., to use patches of size 1-by-1, for prediction of deformation? This answer is NO due to the *aperture problem* accompanied with patch-based representation. Suppose we have a template image I_0 and a region R_0 (Fig. 5.1). If a small deformation field moves the image content within R_0 slightly to form

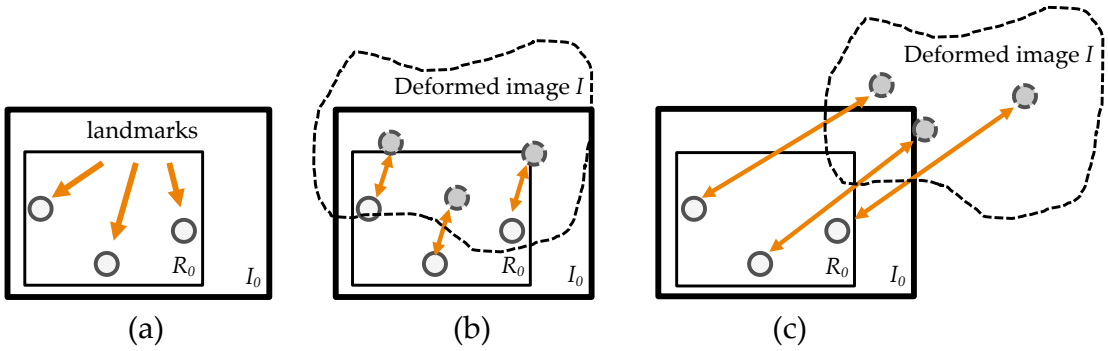


Figure 5.1: Aperture Problem. **(a)** A patch R_0 on the template image I_0 with three landmarks. **(b)** When deformation is small, the feature extracted from R_0 can be used to predict local landmark displacements. **(c)** When deformation is large, the feature extracted from R_0 is no longer related to the local deformation.

a deformed image I, then by extracting information from R_0 in I (namely the rectangle patch $I(R_0)$), one can still predict the small deformation quite accurately. However, if the deformation is large and moves the template content $I_0(R_0)$ to another region in I, then a prediction using $I(R_0)$ as the input is not expected to perform well.

In summary, a patch-based representation is double-bladed:

- **Positive:** Deformation on a patch has lower dimensionality and fewer degrees of freedom, which substantially reduces the sample complexity.
- **Negative:** Large deformation cannot be predicted from small patches due to aperture problem.

A question naturally arises: how to avoid aperture problems while enjoying fewer degrees of freedom from patch-based representation?

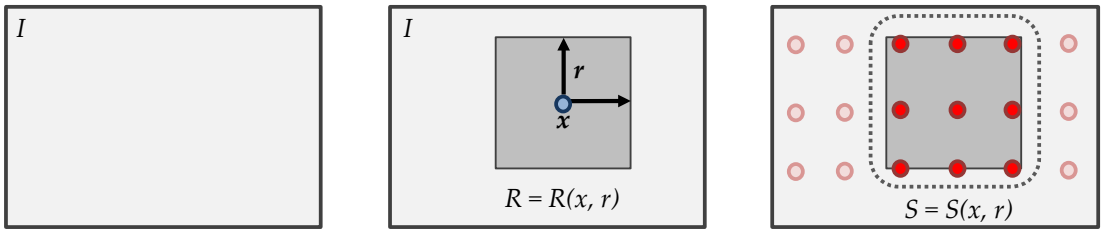
The answer proposed in this thesis is to use a *hierarchical structure*. The second part of the thesis will show how to construct a hierarchy that predicts large and complex deformations using local patches of different sizes in a principled manner. As pointed out by many previous works, there exist two ways to operate on such a hierarchy, namely *top-down* and *bottom-up*. Accordingly, two approaches, Top-Down and Bottom-Up Hierarchical Predictions, are proposed in a principled way with global optimality guarantees and even lower sample complexity.

5.1 Local Reparameterization of Warping

Let us first rephrase the problem, and introduce basic notations for part-based representation. Given a template I_0, a deformed image $I_{\mathbf{p}}$ is generated from a deformation field $W(\mathbf{x};\mathbf{p})$. The

Subscripts Usage	
i	(Training) sample index.
j	Patch index or an index into a parent vertex in hierarchical models.
k	Landmark index, or index into a child vertex in hierarchical models.
Hierarchy	
T	Number of layers
t	Layer number
$[t]$	All parts in layer t
$ch(j)$	Children set of part j.
$ah(j)$	Anchor index of part j.
h_j	Held state for part j. $h_j = (\mathbf{u}_j, z_j)$, where $\mathbf{u}_j$ is the 2D location and z_j the type variable.
g_j	Discarded state for part j.
$z_k \sim z_j$	Type variable z_k of child k is compatible with type variable z_j of parent j.
$h_k \sim (h_j, g_j)$	The child state h_k is compatible with the parent state (h_j, g_j).
Patch/Part j	
R_{j0}	Rectangle/Region of patch j on template image I_0.
R_j	Rectangle/Region on deformed image. $R_j = R_j(\mathbf{p})$ is a function of parameter $\mathbf{p}$.
S_j	Subset of dominant landmarks in patch j
r_j	Characteristic range of part j
ϵ_j	Inverse of sample density related to patch j in bottom-up hierarchy.
$\mathbf{p}(S_j)$	Subset of dominant parameters in patch j
$I_{\mathbf{p}}(R_j(\mathbf{q}))$	Local pull-back operation.
Lipschitz Conditions	
α_j	(Inverse of) density of training samples.
γ_j	(One minus) Convergence rate.
A_j, Γ_j	Lipschitz constants.

Table 5.1: Additional Notations used in Patch-based Modeling.



(a) Image I (b) Image Patch $I(R)$ (c) The subset S of landmarks associated with $I(R)$

Figure 5.2: Illustration of $I(R)$ and subset S of landmarks.

deformed image $I_{\mathbf{p}}$ satisfies the following equation:

$$I_{\mathbf{p}}(W(\mathbf{x};\mathbf{p})) = I_0 \tag{5.1}$$

In the first part of the thesis, we assume $W(\mathbf{x};\mathbf{p})$ as a linear combination of a set of *global* bases $B_G(\mathbf{x})$ with *global* parameters $\mathbf{p}_G$ (See Sec. 3.2):

$$W(\mathbf{x};\mathbf{p}) = \mathbf{x} + B_G(\mathbf{x})\mathbf{p}_G \tag{5.2}$$

In this chapter, instead of using global parameters $\mathbf{p}_G$, we use a local deformation model. From the template I_0, the deformed image $I_{\mathbf{p}}$ is generated by moving around K uniformly distributed *landmarks* $\{\mathbf{l}_k\}_{k=1}^K$ on the template, and interpolate other pixel displacements accordingly.

In this setting, we can write Eqn. 5.2 in the following *local* form:

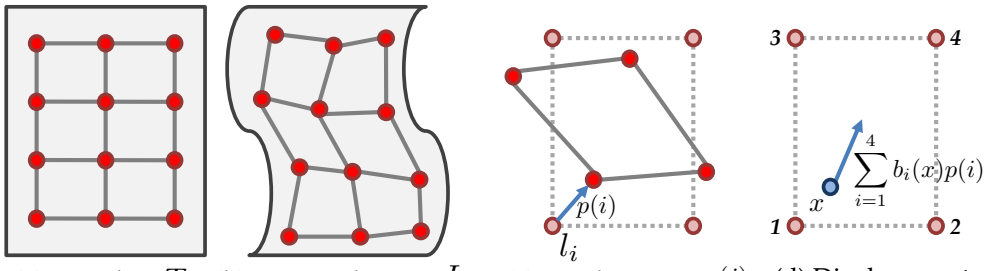
$$W(\mathbf{x};\mathbf{p}) = \mathbf{x} + (B_L(\mathbf{x})\mathbf{p}_L)^\top \tag{5.3}$$

where the bases $B_L(\mathbf{x}) = [b_1(\mathbf{x}), b_2(\mathbf{x}), \ldots, b_K(\mathbf{x})]$ is a K-dimensional row vector of weighting factors on location $\mathbf{x}$ from K landmarks and $\mathbf{p}_L$ is a K-by-2 matrix storing all the displacements of landmarks. For a specific k, $\mathbf{p}_L(k)$ is a 2-dimensional row vector, which is the displacement of k-th landmark.

5.1.1 Property of bases $B(\mathbf{x})$.

Mathematically, $\{b_k(\mathbf{x})\}_{k=1}^K$ are *local* in the sense that $b_k(\mathbf{x})$ is high when $\mathbf{x}$ is close to k-th landmark $\mathbf{l}_k$, but decays quickly otherwise. At any location $\mathbf{x}$, its weights from all K landmarks add to 1:

$$\sum_k b_k(\mathbf{x}) = 1 \qquad \forall \mathbf{x} \tag{5.4}$$



(a) Template T (b) Distorted Image I_p (c) Displacement $p(i)$ for landmark l_i (d) Displacement at pixel x

Figure 5.3: Local parameterization of deformation. **(a)-(b)** The deformation field is controlled by a set of landmarks on the template image. By moving these landmarks, a deformed image is created. **(c)** Local parameterization. Each parameter $\mathbf{p}(k)$ encodes the 2D displacement of the landmark k. **(d)** Displacement on any pixel $\mathbf{x}$ is interpolated using displacements of nearby landmarks.

and the weight $b_k(\cdot)$ at k-th landmark location $\mathbf{l}_k$ is 1 while others are zero:

$$b_k(\mathbf{l}_k) = 1, \qquad b_j(\mathbf{l}_k) = 0 \quad \forall j \neq k \tag{5.5}$$

Practically, $B_L(\mathbf{x})$ can be any interpolation function, e.g., Radial Basis Function, Thin-plate Spline [10], B-spline [69], local linear interpolation, etc. Note for some of those choices, Eqn. 5.4 may not hold and a normalization procedure is needed.

We assume that $B(\mathbf{x}) = [b_1(\mathbf{x}), b_2(\mathbf{x}), \ldots, b_K(\mathbf{x})]$ is smoothly changing:

Assumption 5.1.1 *There exists c_B so that:*

$$\|(B(\mathbf{x}) - B(\mathbf{y}))\mathbf{p}\|_\infty \leq c_B \|\mathbf{x} - \mathbf{y}\|_\infty \|\mathbf{p}\|_\infty \tag{5.6}$$

Intuitively, Eqn. 5.6 measures how smooth the bases change over space.

Lemma 5.1.2 (Unity bound) *For any $\mathbf{x}$ and any $\mathbf{p}$, we have $\|B(\mathbf{x})\mathbf{p}\|_\infty \leq \|\mathbf{p}\|_\infty$.*

Proof

$$\|B(\mathbf{x})\mathbf{p}\|_\infty = \max\{\sum_i b_i(\mathbf{x})\mathbf{p}^x(i), \sum_i \mathbf{b}_i(\mathbf{x})\mathbf{p}^y(i)\} \tag{5.7}$$

$$\leq \max\{\max_i \mathbf{p}^x(i) \sum_i b_i(\mathbf{x}), \max_i \mathbf{p}^y(i) \sum_i b_i(\mathbf{x})\} = \|\mathbf{p}\|_\infty \tag{5.8}$$

using the fact that $\sum_i b_i(\mathbf{x}) = 1$ for any $\mathbf{x}$. ∎

5.1.2 Relationship between global and local parameterization

Local overparameterization can be reformulated to a global one:

$$W(\mathbf{x};\mathbf{p}) = \mathbf{x} + (\underbrace{B_L(\mathbf{x})}_{\text{1-by-K}} \underbrace{\mathbf{p}_L}_{\text{K-by-2}})^\top \tag{5.9}$$

$$= \mathbf{x} + \underbrace{\begin{bmatrix} b_1(\mathbf{x}) & \dots & b_K(\mathbf{x}) & 0 & 0 & 0 \\ 0 & 0 & 0 & b_1(\mathbf{x}) & \dots & b_K(\mathbf{x}) \end{bmatrix}}_{B_G(\mathbf{x}),\ \text{2-by-}2K} \underbrace{\begin{bmatrix} \mathbf{p}^x(1) \\ \dots \\ \mathbf{p}^x(K) \\ \mathbf{p}^y(1) \\ \dots \\ \mathbf{p}^y(K) \end{bmatrix}}_{\mathbf{p}_G,\ 2K\text{-by-1}} \tag{5.10}$$

From this, it is clear that a local parameterization with K landmarks corresponds to $D = 2K$ bases in the global parameterization case. However, because bases and displacements from nearby landmarks are highly correlated, $\{b_k(\mathbf{x})\}_{k=1}^K$ are not orthogonal and the effective degrees of freedom $d \ll D$. For this *overcomplete* set of bases, one can reduce D by a Gram-Schmidt orthogonalization process. However, as we shall see in Sec. 6, working with an overcomplete set of bases could be more convenient and substantially reduce the training samples needed.

5.2 Patch-based Representation

5.2.1 Patches on Template I_0

We first discuss the representation for patches on template I_0. For each patch j, denote R_{j0} as a region on I_0. The subscript 0 means the location of the region is fixed when the deformation parameters $\mathbf{p}$ change. That is why R_{j0} is called "on I_0". The shape of the region can be a square, a rectangle or a circle. In all cases, denote r_j as the radius of R_{j0}. For example, for squared region R_{j0}:

$$R_{j0}(\mathbf{x}_j, r_j) = \{\mathbf{x} : \|\mathbf{x} - \mathbf{x}_j\|_\infty \le r_j\} \tag{5.11}$$

and for circular region:

$$R_{j0}(\mathbf{x}_j, r_j) = \{\mathbf{x} : \|\mathbf{x} - \mathbf{x}_j\|_2 \le r_j\} \tag{5.12}$$

However, a region may not be defined exclusively by its center $\mathbf{x}_j$ and its radius r_j. Alternatively, R_{j0} can be defined by its four corners, etc.

Given an arbitrary image I, $I(R_{j0})$ is the patch extracted from an image I, and can be regarded as a infinite-dimensional vector:

$$I(R_{j0}) = \{I(\mathbf{x}) : \mathbf{x} \in R_{j0}\} \tag{5.13}$$

When R_{j0} is sampled in a discrete manner (e.g., sample 10x10 grid on a 30x30 region), $I(R_{j0})$ is also a finite-dimensional vector. The most common case is *sampling per pixel*, i.e., for a 30-by-30 region, $I(R_{j0})$ is a 900-dimensional vector.

5.2.2 Dominant Landmarks and Local Degrees of Freedom

On a region R_{j0}, S_j is defined as *dominant subset of landmarks* whose displacements $\mathbf{p}(S_j)$ *dominantly* influence the patch content $I(R_{j0})$. Here $\mathbf{p}(S_j)$ is a $|S_j|$ by 2 matrix obtained by choosing S_j rows from the (local) parameters $\mathbf{p}$ in Eqn. 5.3. The subscript "L" is omitted for clarity.

Of course, due to the aperture problem, if the displacements are large enough, then the effect of $\mathbf{p}(S_j)$ on $I(R_{j0})$ will diminish and the displacements from other landmarks will take over. Therefore, such a dominance is only valid within the radius r_j so that $\|\mathbf{p}S_j\|_\infty \leq r_j$. In Sec. 6.1, the term "dominance" will be rigorously defined by a relaxed version of Lipschitz condition. In practice, often the landmarks that is within and close to R_{j0} are selected as members in S_j:

$$S_j = \{k : \text{dist}(\mathbf{l}_k, R_{j0}) \leq \text{margin}\} \tag{5.14}$$

See Fig. 5.2 for illustration.

By definition, the degree of freedom in the patch $I(R_{j0})$ is determined by the subset of landmarks S_j. Since $\mathbf{p}(S_j)$ is a $|S_j|$-by-2 matrix, there are $d_j \equiv 2|S_j|$ apparent degrees of freedom.

If the landmarks are uniformly distributed on the 2D image, which could be trivially made true by placing the landmarks on a regular grid, then the number of landmarks is proportional to the area of R_{j0}, which is proportional to r_j^2 since images are two-dimensional. Thus, we conclude that $d_j \equiv 2|S_j| = \beta r_j^2$ for some positive constant β. β is related to the overall smoothness of the deformation field.

On the other hand, for large patch that contains many landmarks, $2|S_j|$ could be huge. However, the overall effective degree of freedom is d. Therefore, if $d < 2|S_j|$, then $\mathbf{p}(S_j)$ contains dependent displacements and the effective degree of freedom in R_{j0} is d.

Let me give a concrete example. If an image undergoes an affine transform that has 6 degrees

of freedom, then even if the number of landmarks $K = 100$, the overall degrees of freedom is always 6. Moreover, for any region R_{j0} that covers $|S_j|$ landmarks with $|S| \geq 3$, the local deformation on the region R_{j0} is still an affine transform, and its degrees of freedom never exceed 6.

Combining these two observations, we thus arrive at the following assumption:

Assumption 5.2.1 (Degrees of Freedom for Patches) *For j-th patch, its local degrees of freedom $d_j = \min(d, 2|S_j|) = \min(d, \beta r_j^2)$.*

5.3 Local Pull-back Operation

5.3.1 Patches on Deformed Image

Following Sec. 5.2, similarly, we can define region R_j on a deformed image. Here the subscript 0 is omitted since the region $R_j = R_j(\mathbf{p})$ moves with the deformation parameters $\mathbf{p}$. Mathematically, R_j is the image of R_{j0} under deformation field $W(\mathbf{x}; \mathbf{p})$:

$$R_j = W(R_{j0}; \mathbf{p}) = \{\mathbf{y} : \mathbf{y} = W(\mathbf{x}; \mathbf{p}),\ \mathbf{x} \in R_{j0}\} \tag{5.15}$$

Under finite resolution, R_j contains a list of pixel locations and both R_j and R_{j0} contain the same number of pixels.

For local deformations controlled by K landmarks, R_j is not necessarily dependent on all the landmarks but only the dominant subset S_j, i.e., $R_j = R_j(\mathbf{p}(S_j))$. For example, if a rectangle region R_{j0} contains 4 dominant landmarks on its four corners, then the location of shape of its deformed rectangle R_j is (almost) determined by the location of the four corners, with a few wiggles on the edges (See Fig. 5.4). Moreover, when landmarks become denser, such a dominant relationship becomes stronger.

For the bottom-up hierarchical model (Sec. 7), we use a low-dimensional projection of $\mathbf{p}(S_j)$ rather than $\mathbf{p}(S_j)$ itself to compute the location and shape of the deformed region R_j. The reason is that if S_j covers a large region, then $\mathbf{p}(S_j)$ is high-dimensional and may go beyond computational complexity available.

5.3.2 Local Pull-back Operation

For arbitrary image I and a moving region $R_j = R_j(\mathbf{q})$, we define $I(R_j)$ as the *local pull-back*:

$$I(R_j(\mathbf{q})) \equiv I(W(R_{j0}; \mathbf{q})) \tag{5.16}$$

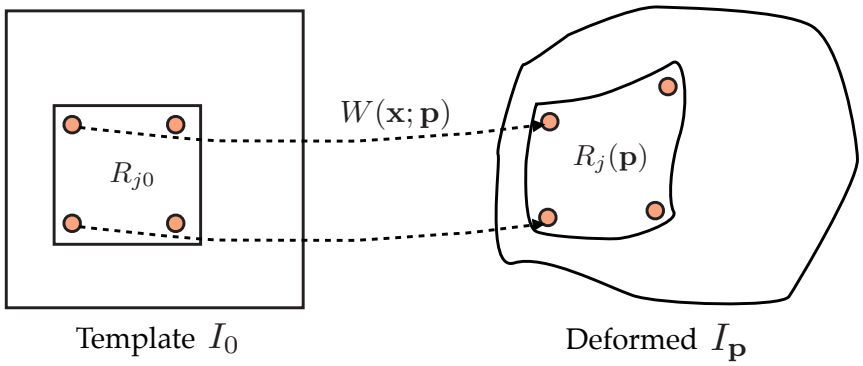


Figure 5.4: Deformed region R_j as a function of $\mathbf{p}$.

In particular, for deformed image $I_{\mathbf{q}}$ and the moving region $R_j = R_j(\mathbf{q})$, we have

$$I_{\mathbf{q}}(R_j(\mathbf{q})) = I_{\mathbf{q}}(W(R_{j0};\mathbf{q})) = I_0(R_{j0}) \tag{5.17}$$

which gives back the template content. This coincides with the (global) pull-back property in Eqn. 3.9. This similarity is the underlying motivation for calling Eqn. 5.16 "local pull-back operation".

From Eqn.5.16, there is a relationship between the (global) pull-back operation $H(I,\mathbf{q})$ and the local pull-back operation $I(R_j(\mathbf{q}))$:

$$H(I,\mathbf{q})(R_{j0}) = I(W(R_{j0};\mathbf{p})) = I(R_j(\mathbf{q})) \tag{5.18}$$

Therefore, to compute $I(R_j(\mathbf{q}))$ for all patches, we do not need to apply Eqn. 5.16 one patch after another. Rather, we can compute the global pull-back image $H(I,\mathbf{q})$ once and extract region R_{j0} for j-th patch on the pull-back image.

5.3.3 Pull-back Inequality

Similar to the pull-back inequality for the entire image (Eqn. 3.10):

$$\|H(I_{\mathbf{p}},\mathbf{q}) - I_{\mathbf{p}-\mathbf{q}}\| \leq c_H\|\mathbf{p}-\mathbf{q}\|_\infty, \tag{5.19}$$

we have the following local pull-back inequality:

Theorem 5.3.1 *For j-th patch with region R_j and radius r_j, if $\|\mathbf{p}-\mathbf{q}\|_\infty \leq r_j$ and $\|\mathbf{q}\|_\infty \leq c_q$, then*

$$\|I_{\mathbf{p}}(R_j(\mathbf{q})) - I_{\mathbf{p}-\mathbf{q}}(R_{j0})\| \leq \eta_j r_j \tag{5.20}$$

where $\eta_j = c_B c_q c_G Area_j$. *Note* $c_G = \max_{\mathbf{x}} |\nabla I_{\mathbf{p}}(\mathbf{x})|_1$.

Proof For any $\mathbf{y} \in R_{j0}$, by definitions of Eqn. 5.16 and Eqn. 3.3, we have:

$$\begin{aligned} I_{\mathbf{p}}(R_j(\mathbf{q}))(\mathbf{y}) &= I_{\mathbf{p}}(W(\mathbf{y};\mathbf{q})) && (5.21)\\ I_{\mathbf{p}-\mathbf{q}}(\mathbf{y}) &= I_0(W^{-1}(\mathbf{y};\mathbf{p}-\mathbf{q})) = I_{\mathbf{p}}(W(W^{-1}(\mathbf{y};\mathbf{p}-\mathbf{q}),\mathbf{p})) && (5.22)\end{aligned}$$

Now we need to check the pixel distance between $\mathbf{u} = W(\mathbf{y};\mathbf{q})$ and $\mathbf{v} = W(W^{-1}(\mathbf{y};\mathbf{p}-\mathbf{q}),\mathbf{p})$. Note both are pixel locations on distorted image $I_{\mathbf{p}}$. If we can bound $\|\mathbf{u}-\mathbf{v}\|_\infty$, then from $I_{\mathbf{p}}$'s appearance, we can obtain the bound for $|I_{\mathbf{p}}(R_j(\mathbf{q}))(\mathbf{y}) - I_{\mathbf{p}-\mathbf{q}}(\mathbf{y})|$.

Denote $\mathbf{z} = W^{-1}(\mathbf{y};\mathbf{p}-\mathbf{q})$ which is a pixel on the template. By definition we have:

$$\mathbf{y} = \mathbf{z} + B(\mathbf{z})(\mathbf{p}-\mathbf{q}) \tag{5.23}$$

then we have $\|\mathbf{y}-\mathbf{z}\|_\infty = \|B(\mathbf{z})(\mathbf{p}-\mathbf{q})\|_\infty \le \|\mathbf{p}-\mathbf{q}\|_\infty \le r_j$ by Lemma 5.1.2. On the other hand, we have:

$$\begin{aligned} \mathbf{u}-\mathbf{v} &= W(\mathbf{y},\mathbf{q}) - W(\mathbf{z},\mathbf{p}) && (5.24)\\ &= \mathbf{y} + B(\mathbf{y})\mathbf{q} - \mathbf{z} - B(\mathbf{z})\mathbf{p} && (5.25)\\ &= B(\mathbf{z})(\mathbf{p}-\mathbf{q}) - B(\mathbf{z})\mathbf{p} + B(\mathbf{y})\mathbf{q} && (5.26)\\ &= (B(\mathbf{y}) - B(\mathbf{z}))\mathbf{q} && (5.27)\end{aligned}$$

Thus, from Eqn. 5.6 we have:

$$\|\mathbf{u}-\mathbf{v}\|_\infty \le c_B \|\mathbf{y}-\mathbf{z}\|_\infty \|\mathbf{q}\|_\infty \le (c_B\|\mathbf{q}\|_\infty) r_j \tag{5.28}$$

Thus:

$$\begin{aligned} |I_{\mathbf{p}}(R_j(\mathbf{q}))(\mathbf{y}) - I_{\mathbf{p}-\mathbf{q}}(\mathbf{y})| &= |I_{\mathbf{p}}(W(\mathbf{y};\mathbf{q})) - I_{\mathbf{p}}(W(W^{-1}(\mathbf{y};\mathbf{p}-\mathbf{q}),\mathbf{p}))| && (5.29)\\ &= |I_{\mathbf{p}}(\mathbf{u}) - I_{\mathbf{p}}(\mathbf{v})| && (5.30)\\ &\le |\nabla I_{\mathbf{p}}(\xi)|_1 \|\mathbf{u}-\mathbf{v}\|_\infty && (5.31)\\ &\le c_B |\nabla I_{\mathbf{p}}(\xi)|_1 \|\mathbf{q}\|_\infty r_j && (5.32)\end{aligned}$$

where $\xi \in$ Line-Seg$(\mathbf{u},\mathbf{v})$. Collecting Eqn. 5.32 over the entire region R_j gives the bound. ∎

Practically, η_j is very small and can be neglected.

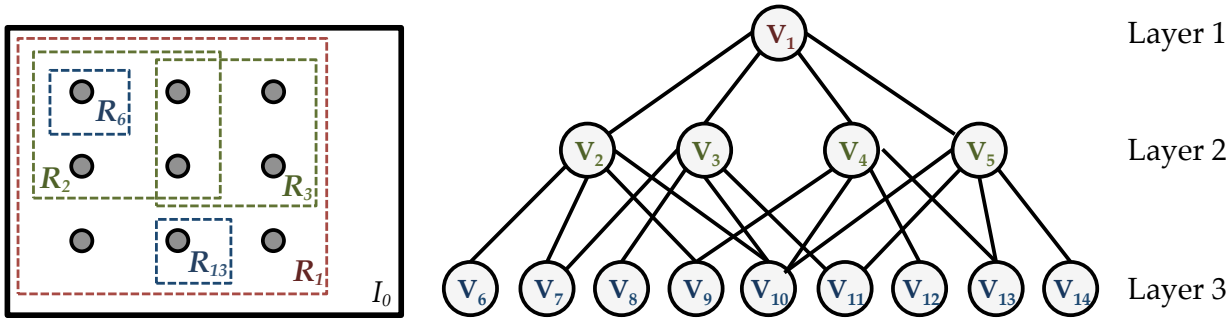


Figure 5.5: An example of the hierarchical structure.

5.4 From Patches to Hierarchy

From all the patches, now we are ready to build a hierarchical structure for deformation estimation. Following different principles, i.e., top-down or bottom-up, the criteria to build the hierarchy are also different. I leave the principles and details to Chapter 6 and Chapter 7. Here I will introduce common basic notations for the hierarchy.

The entire hierarchy can be regarded as a graph. The j-th patch corresponds to vertex V_j. Two vertices V_j and V_k is a *parent-child pair* with V_j being a parent and V_k being a child, if the two dominant subsets satisfy $S_j \supseteq S_k$. In such a case, there is an edge between the two vertices. For any node V_j, denote $ch(V_j)$ as its children, and $pa(V_j)$ as its parents. In abbreviation, we also use $ch(j)$ or ch_j to represent $ch(V_j)$.

All vertices in the hierarchy are arranged in a layer-by-layer fashion. For each vertex V_j, there is an associate layer number $t = t(V_j)$. The notation $[t]$ is the set of all the nodes that belong to layer t. For a parent-child pair, the parent V_j is always one layer higher than the child V_k, i.e., $t(V_j) = t(V_k) - 1$. The root node whose dominant subset $S_j = \{1, 2, \ldots, K\}$ covers all landmarks has $t = 1$, while the leaf node whose $|S_j| = 1$ has $t = T$. The total number of layers are T.

Regions (vertices) in the same layer share the same size of dominant subsets and the same scale. This leads to two consequences: first, two distinct nodes V_{j1} and V_{j2} in the same layer with $S_{j1} \neq S_{j2}$ cannot be connected (no edges in the same layer). We denote the common size of S as $|S|_t$. Second, the scale of regions is fixed within layer, and is a function of layer number $t(j)$, denoted as r_t. $r_t > r_{t+1}$ since region in the higher layer has a larger scale compared to the regions in the lower layer. Note that in this section, we do not specify the shrinking factor r_{t+1}/r_t. It depends on the specific design criterion of the algorithm and will be elaborated later.

Fig. 5.5 shows an example of hierarchy. V_1 is the root node with $t(1) = 1$ and V_6-V_{14} are the

leaf nodes with $t(V_6) = t(V_7) = \ldots = t(V_{14}) = 3$. $ch(V_1) = \{2, 3, 4, 5\}$, $ch(V_2) = \{6, 7, 8, 9\}$ and $pa(V_{10}) = \{2, 3, 4, 5\}$. Note one node may have more than one parents. So in general, the hierarchy is not a tree.

Chapter 6

Top-Down Hierarchical Prediction: Factorizing Deformation onto Patches

As shown in Sec. 4, the Data-driven Descent algorithm reduces the sample complexity from $O(1/\epsilon^d)$ to $O(C^d \log 1/\epsilon)$, while maintaining the same global optimality guarantee: $\|\hat{\mathbf{p}} - \mathbf{p}\|_\infty \leq \epsilon$. Intuitively, this approach builds links of two deformed images with the compositional structure of deformation, despite they are far away from each other in terms of image metric. Therefore, a test image can leverage the training samples which are far away in appearance but connected by deformation composition, for its prediction. It shows good empirical results for local deformation, but fails to capture general deformation that contains both global and local components (e.g., cloth moving and deforming). Indeed, in Sec. 4.7, manual initialization has to be introduced to remove the global components of cloth deformation.

In this chapter, Top-Down Hierarchical Predictions are proposed for deformation that contains both global and local components. Still it comes with global optimality guarantee.

The main intuition is as follows. First, the Lipschitz conditions between the image content and the deformation parameters (Eqn. 3.22) are relaxed to deal with noise and patch boundaries. With the relaxed conditions assumed on all patches of different locations and scales, every patch can be regarded as predictors with guaranteed worst-case precisions of different granularity. Patches with large scales can deal with large deformation but the precision is low, while patches with small scales only deal with small deformation but with high precision.

Because of their complementary properties, patches of different scales can be concatenated. Patches of large scale can deal with large deformation first, output a coarse estimation and reduce the estimation error to a level, which smaller patches can kick in and further improve the estimation. Thanks to local parameterization structure of the deformation field W by K landmarks

(See Eqn. 5.3):

$$W(\mathbf{x};\mathbf{p}) = \mathbf{x} + (B_L(\mathbf{x})\mathbf{p}_L)^\top, \tag{6.1}$$

deformation that happens within a local image patch (i.e., the dominant subset mentioned in Sec. 5.2.2) can be predicted by the content of that patch. Therefore, deformation on small patches at different locations are conditionally independent of each other once the global deformation has been estimated, with a few samples their performance can be guaranteed. These patch predictors are thus cascaded together in a top-down hierarchical manner, able to handle large and high-dimensional deformation with both local and global components.

As a result, compared to Data-Driven Descent, Top-Down Hierarchical Prediction brings down sample complexity to $O(C_1^d + C_2 \log 1/\epsilon)$ where d is the degrees of freedom in Eqn. 6.1. This sample complexity varies very slowly with respect to the accuracy. In particular, the number of samples required in each iteration stays constant for the first few layers of hierarchy, and then decays *double exponentially*. Furthermore, the sample complexity guarantee is based on relxed Lipschitz conditions that can be verified with an efficient algorithm. This reduces the constant C in Eqn. 4.13 to a much smaller constant C_1.

Practically, Top-Down Hierarchical Prediction also demonstrates good quantitative and qualitative results on real video sequences containing different types of deformation, including clothing and water surface deformations as well as medical images of internal organs. This approach outperforms optimization-based approaches such as Lucas-Kanade [6] and Free-form registration [69] (both with coarse-to-fine implementations), regression-based approaches such as Nearest Neighbor and Explicit Shape Regression [14], feature-based approaches such as SIFT [53], tracking-based approaches such as KLT [83], and finally previous proposed Data-Driven Descent. Currently the unoptimized Matlab implementation is fast, achieving 3-4 fps on real images.

6.1 Relationship between Local Parameters and Local Image Appearance

6.1.1 Motivation

One shortcoming of the Lipschitz condition proposed in Sec. 3.6.1:

$$L_1\Delta I \leq \Delta\mathbf{p} \leq L_2\Delta I \qquad \forall \mathbf{p}_1, \mathbf{p}_2 : \|\mathbf{p}_1\|_\infty, \|\mathbf{p}_2\|_\infty \leq r_0 \tag{6.2}$$

is that it must hold for arbitrarily small ΔI and $\Delta\mathbf{p}$. Thus it fails in the following two situations:

- **Noisy images.** Adding noise to a distorted image I_p changes its appearance but not its parameters. As a result, $\Delta\mathbf{p} \approx 0$ but ΔI is finite. This makes $L_1 \to 0$.
- **Repetitive Patterns.** If an image resembles itself after some transformation, then $\Delta\mathbf{p}$ is finite but $\Delta I \approx 0$. This makes $L_2 \to +\infty$.

In both cases, the analysis in Sec. 4.2 gives a trivial (infinite) bound on sample complexity and global optimality cannot be guaranteed.

6.1.2 Relaxed Lipschitz Conditions

With patch-based representation introduced in Sec. 5.2, the global Lipschitz condition (Eqn. 6.2) is replaced with a patch-wise *relaxed Lipschitz condition* between the image content $I_{\mathbf{p}}(R_{j0})$ and the dominant parameters $\mathbf{p}(S_j)$ of that patch. Recall that R_{j0} is the region of template for j-th patch and S_j is the dominant subset of landmarks in j-th patch.

Assumption 6.1.1 (Relaxed Lipschitz Condition for Patch j) *For patch j with scale r_j and pull-back error η_j, there exists 4-tuples $(\alpha_j, \gamma_j, A_j, \Gamma_j)$ with $0 < \alpha_j \leq \gamma_j < 1$ and $A_j + \eta_j < \Gamma_j$ so that for any $\mathbf{p}_1$ and $\mathbf{p}_2$ with $\|\mathbf{p}_1\|_\infty \leq r_j$, $\|\mathbf{p}_2\|_\infty \leq r_j$, we have:*

$$\Delta\mathbf{p} \leq \alpha_j r_j \quad \Longrightarrow \quad \Delta I \leq A_j r_j \tag{6.3}$$

$$\Delta\mathbf{p} \geq \gamma_j r_j \quad \Longrightarrow \quad \Delta I \geq \Gamma_j r_j \tag{6.4}$$

for $\Delta\mathbf{p} \equiv \|\mathbf{p}_1(S_j) - \mathbf{p}_2(S_j)\|_\infty$ and $\Delta I \equiv \|I_{\mathbf{p}_1}(R_{j0}) - I_{\mathbf{p}_2}(R_{j0})\|$.

Note that if (α_j, γ_j) satisfies Eqn. 6.3 and Eqn. 6.4, then γ_j could be larger without violating the conditions (note A_j and Γ_j could be different). Thus, given α_j we choose the minimal $\gamma_j = \gamma(\alpha_j)$ that meets the condition (Fig. 6.1(b)).

Different from the Lipschitz conditions (Eqn. 6.2), one important aspect of Eqn. 6.3 and Eqn. 6.4 is that ΔI and $\Delta\mathbf{p}$ are only correlated up to the scale of r_j. This allows the conditions to account for the aperture problem, i.e., when the displacements are large, the parameters on the dominant subset S_j are no longer correlated with the patch content $I(R_{j0})$. The range r_j can be regarded as the *acceptance range* for the patch. As a result, the local over-parameterization enables us to consider only a subset S_j of deformations.

The relaxed conditions also account for the noisy correlation case, in which two slightly different parameters share the same image appearance, or the same parameters gives slightly different image appearance. In both situations, the pair (α_j, γ_j) is still well-behaved but $L_2/L_1 \to +\infty$.

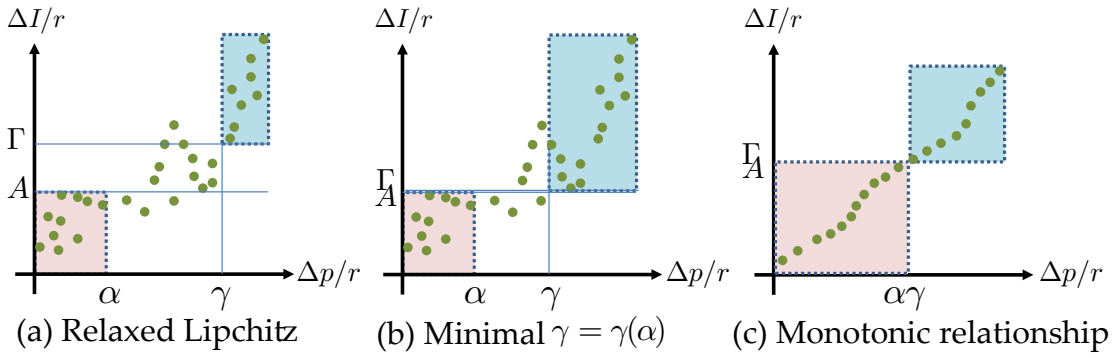


Figure 6.1: Relaxed Lipschitz Condition (Eqn. 6.3 and Eqn. 6.4). **(a)** There are four constants $(\alpha, \gamma, A, \Gamma)$ capturing the correlations between ΔI and $\Delta \mathbf{p}$ when $\Delta \mathbf{p}$ is very small ($\leq \alpha_j r_j$) or very large ($\geq \gamma_j r_j$). **(b)** Minimal γ_j without violating the condition. **(c)** For a monotonic relationship between ΔI and $\Delta \mathbf{p}$, $\alpha_j = \gamma_j \in [0, 1]$.

6.1.3 Empirically Estimation of Lipschitz Constants

The constants in the relaxed Lipschitz conditions (Eqn. 6.3 and Eqn. 6.4) can be empirically estimated from a set of image differences $\{\Delta I_m\}$ and corresponding parameter differences $\{\Delta \mathbf{p}_m\}$, both can be computed from training samples $(\mathbf{p}^{(i)}, I^{(i)})$. Since γ, A, Γ is actually a function of α, what we estimate is a set of plausible 4-tuples, and most importantly, the curve $\gamma = \gamma(\alpha)$. Note that we omit subscript j here for clarity.

For M pairs of image/parameters differences $\{(\Delta \mathbf{p}_m, \Delta I_m)\}$, a brute-force search computes all plausible 4-tuples by enumerating all possible (α, γ) and find feasible ones. This takes $O(M^3)$ operations. Here I propose Alg. 4 which only costs $O(M \log M)$.

Algorithm 4 Find Local Lipschitz Constants

INPUT Parameter distances $\{\Delta \mathbf{p}_m\}$ with $\Delta \mathbf{p}_m \leq \Delta \mathbf{p}_{m+1}$.
INPUT Image distances $\{\Delta I_m\}$.
INPUT Scale r and noise η.
$\Delta I_m^+ = \max_{1 \leq l \leq m} \Delta I_l$, for $i = 1 \dots M$.
$\Delta I_m^- = \min_{i \leq l \leq M} \Delta I_l$, for $i = 1 \dots M$.
for $m = 1$ to M **do**
    Find minimal $l^* = l^*(m)$ so that $\Delta I_{l^*}^- > \Delta I_m^+ + 2\eta$.
    if $m \leq l^*$ **then**
        Store the 4-tuples $(\alpha, \gamma, A, \Gamma) = (\Delta \mathbf{p}_m, \Delta \mathbf{p}_{l^*}, \Delta I_m^+, \Delta I_{l^*}^-)/r$.
    end if
end for

To analyze Alg. 4, we make the following definitions:

Definition 6.1.2 (Allowable set of A and Γ) *Given α, define the allowable set $\tilde{A}(\alpha)$ as:*

$$\tilde{A}(\alpha) = \{A : \forall m\ \Delta\mathbf{p}_m \leq \alpha \implies \Delta I_m \leq A\} \tag{6.5}$$

Naturally we have $\tilde{A}(\alpha') \subset \tilde{A}(\alpha)$ for $\alpha' > \alpha$. Similarly, given γ, define the allowable set $\tilde{\Gamma}(\gamma)$ as:

$$\tilde{\Gamma}(\gamma) = \{\Gamma : \forall m\ \Delta\mathbf{p}_m \geq \gamma \implies \Delta I_m \geq \Gamma\} \tag{6.6}$$

and $\tilde{\Gamma}(\gamma') \subset \tilde{\Gamma}(\gamma)$ for $\gamma' < \gamma$.

Lemma 6.1.3 (Properties of ΔI^+ and ΔI^-) *The two arrays constructed in Alg. 4 satisfy:*

$$\Delta I_m^+ = \min \tilde{A}(\Delta\mathbf{p}_m) \tag{6.7}$$

$$\Delta I_m^- = \max \tilde{\Gamma}(\Delta\mathbf{p}_m) \tag{6.8}$$

Moreover, ΔI_m^+ is ascending while ΔI_m^- is descending with respect to $1 \leq m \leq M$.

Proof (a): First we show $\Delta I_m^+ \in \tilde{A}(\Delta\mathbf{p}_m)$. Since the list $\{\Delta\mathbf{p}_m\}$ was ordered, for any $\Delta\mathbf{p}_l \leq \Delta\mathbf{p}_m$, , we have $l \leq m$. By definition of ΔI_m^+, we have $\Delta I_l \leq \Delta I_m^+$. Thus $\Delta I_m^+ \in \tilde{A}(\Delta\mathbf{p}_m)$.

(b): Then we show for any $A \in \tilde{A}(\Delta\mathbf{p}_m)$, $\Delta I_m^+ \leq A$. For any $1 \leq l \leq m$, since $\Delta\mathbf{p}_l \leq \Delta\mathbf{p}_m$, by the definition of A, we have $\Delta I_l \leq A$, and thus $\Delta I_m^+ = \max_{1\leq l\leq m} \Delta I_l \leq A$.

Therefore, $\Delta I_m^+ = \min \tilde{A}(\Delta\mathbf{p}_m)$. Similarly we can prove $\Delta I_m^- = \max \tilde{\Gamma}(\Delta\mathbf{p}_m)$. ∎

Theorem 6.1.4 *For each $\alpha = \Delta\mathbf{p}_m$, Algorithm 4* without *the check $\alpha \leq \gamma$ always gives the globally optimal solution to the following linear programming:*

$$\min \quad \gamma \tag{6.9}$$

$$\text{s.t.} \quad \Delta I_m \leq A \quad \forall \Delta\mathbf{p}_m \leq \alpha \quad (\textit{or}\ \ A \in \tilde{A}(\alpha)) \tag{6.10}$$

$$\Delta I_m \geq \Gamma \quad \forall \Delta\mathbf{p}_m \geq \gamma \quad (\textit{or}\ \ \Gamma \in \tilde{\Gamma}(\gamma)) \tag{6.11}$$

$$A + 2\eta < \Gamma \tag{6.12}$$

which has at least one feasible solution $(A \to +\infty, \gamma \to -\infty, \Gamma \to -\infty)$ for any α.

Proof Since there are M data points, we can discretize the values of α and γ into M possible values without changing the property of solution.

(a) First we prove every solution given by Alg. 4 (without the final check) is a feasible solution to the optimization (Eqn. 6.9). Indeed, for any $\alpha = \Delta\mathbf{p}_m$, according to Lemma 6.1.3, $A = \Delta I_m^+ \in \tilde{A}(\alpha)$, $\gamma = \Delta\mathbf{p}_{l^*}$, and $\Gamma = \Delta I_{l^*}^- \in \tilde{\Gamma}(\gamma)$ and thus Eqn. 6.10 and Eqn. 6.11 are satisfied. From the construction of Alg. 4, $A + 2\eta < \Gamma$. Thus, the Algorithm 4 gives a feasible solution to Eqn. 6.9.

(b) Then we prove Alg. 4 (without the final check) gives the optimal solution. If there exists $l' < l^*$ so that $\gamma' = \Delta\mathbf{p}_{l'} < \Delta\mathbf{p}_{l^*} = \gamma$ is part of a better solution $(\alpha, \gamma', A', \Gamma')$, then $\tilde{\Gamma}(\gamma') \subset \tilde{\Gamma}(\gamma)$. This means

$$A' + 2\eta < \Gamma' \leq \Delta I^-_{l'} = \max \tilde{\Gamma}(\gamma') \leq \max \tilde{\Gamma}(\gamma) = \Delta I^-_{l^*} \tag{6.13}$$

On the other hand, $A = \Delta I^+_m = \min \tilde{A}(\alpha) \leq A' \in \tilde{A}(\alpha)$. Then, there are two cases:

- $\Delta I^+_m + 2\eta < \Delta I^-_{l'} < \Delta I^-_{l^*}$. This is not possible since the algorithm already find the minimal l^*.
- $\Delta I^+_m + 2\eta < \Delta I^-_{l'} = \Delta I^-_{l^*}$. Then according to the algorithm, $l' = l^*$.

which is a contradiction. ∎

From Theorem 6.1.4, it is thus easy to check that the complete Algorithm 4 (with the check $\alpha \leq \gamma$) gives the optimal pair (α, γ) that satisfies the Relaxed Lipschitz Conditions (Eqn. 6.3 and Eqn. 6.4).

6.2 Guaranteed Prediction using Nearest Neighbor

Now let us study how the relaxed Lipschitz conditions help Nearest Neighbor prediction. We wish to know how well j-th patch can predict the deformation $\mathbf{p}(S_j)$ within its acceptance range r_j (i.e., $\|\mathbf{p}\|_\infty \leq r_j$). For large deformation out of the acceptance range, from relaxed Lipschitz conditions, nothing can be guaranteed and Nearest Neighbor prediction may not work.

Without any training samples, we can trivially set the prediction $\hat{\mathbf{p}}(S_j) = 0$ and get a worst-case guaranteed prediction error of r_j. Now the problem is: if we want to obtain a slightly better prediction, how many training samples do we need?

Theorem 6.2.1 gives the answer. It shows that if the relaxed Lipschitz condition (Eqn. 6.3 and Eqn. 6.4) holds, then a Nearest Neighbor prediction using $1/\alpha_j$ samples per dimension will always reduce the error by a factor of $\gamma_j < 1$:

Theorem 6.2.1 (Guaranteed Nearest Neighbor for Patch j) *Suppose we have an image I so that it is close to a deformed image on the region R_{j0}:*

$$\|I(R_{j0}) - I_{\mathbf{p}}(R_{j0})\| \leq \eta_j r_j \tag{6.14}$$

for some $\mathbf{p}$ with $\|\mathbf{p}\|_\infty \leq r_j$, then with

$$N_j = \min\left(c_{SS}\left\lceil\frac{1}{\alpha_j}\right\rceil^d, \left\lceil\frac{1}{\alpha_j}\right\rceil^{2|S_j|}\right) \tag{6.15}$$

number of samples properly distributed in the hypercube $[-r, r]^{2|S_j|}$*, we can compute a prediction* $\hat{\mathbf{p}}(S_j)$ *so that*

$$\|\hat{\mathbf{p}}(S_j) - \mathbf{p}(S_j)\| \leq \gamma_j r_j \tag{6.16}$$

using Nearest Neighbor in the region R_j *with image metric.*

Proof Since $\|\mathbf{p}\|_\infty \leq r_j$, by definition we have $\|\mathbf{p}(S_j)\|_\infty \leq r_j$ and similarly $\|\mathbf{q}(S_j)\|_\infty \leq r_j$. Then by applying Thm. 13.1.2 and Thm. 13.2.2 with $\alpha = \alpha_j$, if the number of samples needed follows 6.15, then there exists a data sample $\mathbf{q}$ so that its slicing $\mathbf{q}(S_j)$ satisfies:

$$\|\mathbf{p}(S_j) - \mathbf{q}(S_j)\|_\infty \leq \alpha_j r_j \tag{6.17}$$

For $k \notin S_j$, the value of $\mathbf{q}(k)$ is not important as long $\|\mathbf{q}\|_\infty \leq r_j$. This is because by assumption, the relaxed Lipschitz conditions still holds no matter how $\mathbf{q}(S_j)$ is extended to the entire landmark set.

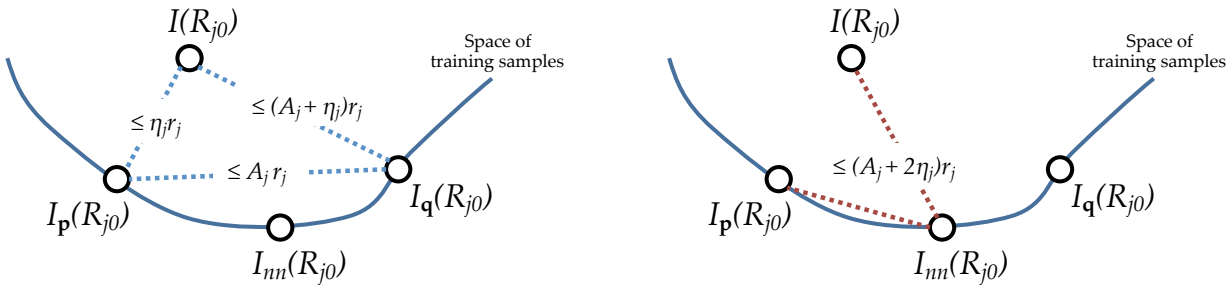


Figure 6.2: Illustration for proof of Guaranteed Nearest Neighbor.

Fig. 6.2 shows the relationship for different quantities involved in the proof. Consider the patch $I_{\mathbf{p}}(R_{j0})$, using Eqn. 6.3 and we have:

$$\|I_{\mathbf{p}}(R_{j0}) - I_{\mathbf{q}}(R_{j0})\| \leq A_j r_j \tag{6.18}$$

Thus we have for the input image I:

$$\|I(R_{j0}) - I_{\mathbf{q}}(R_{j0})\| \leq \|I(R_{j0}) - I_{\mathbf{p}}(R_{j0})\| + \|I_{\mathbf{p}}(R_{j0}) - I_{\mathbf{q}}(R_{j0})\| \leq (A_j + \eta_j) r_j \tag{6.19}$$

On the other hand, since $I_{nn}(R_{j0})$ is the Nearest Neighbor image to I, its distance to I can only be smaller:

$$\|I(R_{j0}) - I_{nn}(R_{j0})\| \leq \|I(R_{j0}) - I_{\mathbf{q}}(R_{j0})\| \leq (A_j + \eta_j) r_j \tag{6.20}$$

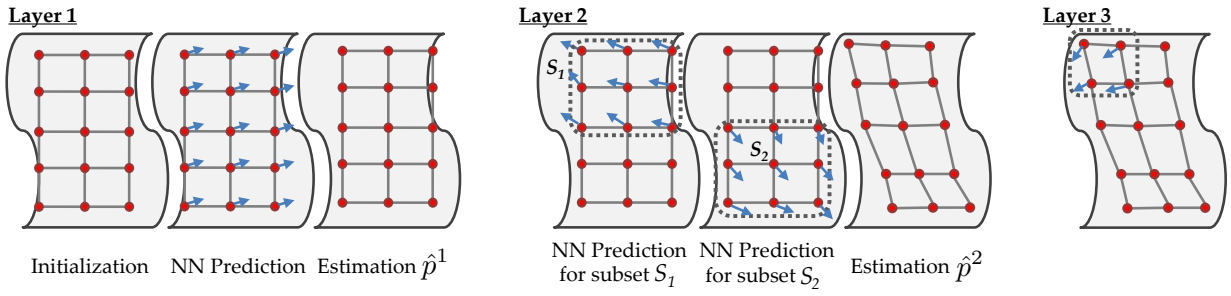


Figure 6.3: Work flow of our hierarchical algorithm for deformation estimation. On Layer 1, a global prediction is made and the estimation is updated. On Layer 2, local deformation is estimated and aggregated. The procedure repeats until the last layer.

Thus we have:

$$\|I_{\mathbf{p}}(R_{j0}) - I_{nn}(R_{j0})\| \leq \|I_{\mathbf{p}}(R_{j0}) - I(R_{j0})\| + \|I(R_{j0}) - I_{nn}(R_{j0})\| \leq (A_j + 2\eta_j)r_j \quad (6.21)$$

Now we want to prove $\|\mathbf{p}(S_j) - \mathbf{q}_{nn}(S_j)\| \leq \gamma_j r_j$. If not, then from Eqn. 6.21 we have:

$$\|I_{\mathbf{p}}(R_{j0}) - I_{nn}(R_{j0})\| \geq \Gamma_j r_j > (A_j + 2\eta_j)r_j \quad (6.22)$$

which from Eqn. 6.4 is a contradiction. Thus we have

$$\|\mathbf{p}(S_j) - \mathbf{q}_{nn}(S_j)\|_\infty \leq \gamma_j r_j \quad (6.23)$$

Thus, just setting the prediction $\hat{\mathbf{p}}(S_j) = \mathbf{q}_{nn}(S_j)$ suffices. ∎

From Theorem 6.2.1, now both α_j and γ_j have their physical meanings: α_j is the inverse of sample complexity per dimension, while γ_j is the inverse of prediction accuracy. Ideally we want α_j to be large for lower sample complexity, and γ_j to be small for higher accuracy. However, the constraint $\alpha_j \leq \gamma_j$ means there is a trade-off. In Sec. 6.4, we show that like L_2/L_1 in Eqn. 3.22, this trade-off reflects the difficulty level of images for deformation prediction.

6.3 Construction of Hierarchical Structure

According to Theorem 6.2.1, different image patches show different characteristics in their prediction guarantees: patches with large r_j can deal with large deformation due to its larger acceptance range, but have low prediction precision, while patches with small r_j only deals with small deformation but enjoys high prediction precision. Therefore, in order to estimate large defor-

mation with high precision, a natural way is to build a coarse-to-fine hierarchy of predictions as follows: the coarse layer (large patches) reduces the prediction residue by a certain extent so that it is within the acceptance range of the fine layer (small patches), where the prediction is refined.

From this argument, we construct the hierarchical structure according to Sec. 5.4. Recall for j-th patch, $t = t(j)$ is the index of its layer. By construction, scale of patches within the same layer t is fixed and denoted as r_t. In this top-down hierarchical predictions, the shrinking factor r_{t+1}/r_t is set to be

$$\bar{\gamma} = \max_j \gamma_j < 1 \tag{6.24}$$

i.e., $r_{t+1} = \bar{\gamma} r_t$.

Algorithm 5 Hierarchical Deformation Estimation.

1: **INPUT** Training samples for j-th patch: $Tr_j \equiv \{\mathbf{p}^{(i)}, I^{(i)}\}$.
2: **INPUT** Test image I_{test} with unknown parameters $\mathbf{p}$.
3: Set an initial estimation $\hat{\mathbf{p}}^0 = 0$.
4: **for** $t = 1$ to T **do**
5: Set the current image $I^t = H(I_{\text{test}}, \hat{\mathbf{p}}^{t-1})$.
6: **for** $j \in [t]$ **do**
7: Find the Nearest Neighbor i^* for patch $I(R_{j0})$:

$$i^* = \arg\min_{i \in Tr_j} \|I^t(R_{j0}) - I^{(i)}(R_{j0})\|$$

8: Set the estimation $\tilde{\mathbf{p}}_j(S_j) = \mathbf{p}^{(i^*)}(S_j)$.
9: **end for**
10: For each landmark k, take average predictions from all the overlapping patches $S[k] = \{j : k \in S_j, j \in [t]\}$:

$$\tilde{\mathbf{p}}(k) = \text{mean}_{j \in S[k]} \tilde{\mathbf{p}}_j(k)$$

11: Update: $\hat{\mathbf{p}}^t = \hat{\mathbf{p}}^{t-1} + \tilde{\mathbf{p}}$.
12: **end for**
13: Return final predictions $\hat{\mathbf{p}}$ for all landmarks.

Fig. 6.3 and Alg. 5 illustrate the algorithm that estimates the unknown parameter $\mathbf{p}$ given the test image I_{test}. For the first iteration, the test image I_{test} is directly compared with the training samples generated from the entire image with scale r_1 to obtain the Nearest Neighbor prediction $\hat{\mathbf{p}}^1$. Then for the second iteration, we have a slightly less distorted image $I^2 = H(I_{\text{test}}, \hat{\mathbf{p}}^1) = I_{\text{test}}(W(\mathbf{x}, \hat{\mathbf{p}}^1))$, from which we estimate $\mathbf{p} - \hat{\mathbf{p}}^1$. Since $\|\mathbf{p} - \hat{\mathbf{p}}^1\|_\infty$ is smaller than $\|\mathbf{p}\|_\infty$, its predictions can be localized to smaller patches. Then this procedure is iterated until the lowest level is reached.

Similar to Data-driven Descent, this algorithm will converge to the globally optimal solution, as shown in Theorem 6.3.1. This is achieved by reducing the residue by a factor of $\bar{\gamma}$ in each iteration. It turns out that the number of samples needed for layer t goes down *double-exponentially* with respect to t. This is because if we follow the hierarchy from large patches to small patches, the degrees of freedom d_j of patches stays the same until $d_j \approx d$, and then goes down exponentially (by $\bar{\gamma}^2$) since the number of landmarks within each patch goes down exponentially (See Assumption. 5.2.1). As a result, the required number of samples is $O(C_1^d + C_2 \log 1/\epsilon)$, as shown in Theorem 6.3.2.

Theorem 6.3.1 (The Global Convergence Theorem) *If $\|\mathbf{p}\|_\infty \leq r_1$, then the prediction $\hat{\mathbf{p}}^t$ satisfies:*

$$\|\hat{\mathbf{p}}^t - \mathbf{p}\|_\infty \leq \bar{\gamma}^t r_1 \tag{6.25}$$

As a result, the final prediction $\hat{\mathbf{p}}^T$ satisfies:

$$\|\hat{\mathbf{p}}^T - \mathbf{p}\|_\infty \leq \bar{\gamma}^T r_1 \to 0 \tag{6.26}$$

for sufficiently deep structure $T \to +\infty$.

Proof First from the proof of Thm. 6.2.1, we can see the prediction $\tilde{\mathbf{p}}_j$ given by j-th patch satisfies $\|\tilde{\mathbf{p}}_j\|_\infty \leq r_t$ for $j \in [t]$, since the prediction is picked from a hypercube $[-r_t, r_t]^{2|S_j|}$. Then, for the overall prediction $\hat{\mathbf{p}}^t$ of any t, we have:

$$\|\hat{\mathbf{p}}^t\|_\infty = \|\sum_{l=1}^{t-1} \tilde{\mathbf{p}}^l\|_\infty \leq \sum_{l=1}^{t-1} \|\tilde{\mathbf{p}}^l\|_\infty \leq \sum_{l=1}^{t-1} r_l \leq \sum_{l=1}^{+\infty} r_l \leq \frac{r_1}{1-\bar{\gamma}} \tag{6.27}$$

Then we prove the main result by induction. Suppose after layer t is processed, the residue $\delta\mathbf{p}^t \equiv \mathbf{p} - \hat{\mathbf{p}}^t$ satisfies:

$$\|\delta\mathbf{p}^t\|_\infty \leq r_{t+1} \tag{6.28}$$

This is trivially true for $t = 0$ by the premise $\|\mathbf{p}\|_\infty \leq r_1$ and initialization of the algorithm $\hat{\mathbf{p}}^0 = 0$. Now suppose Eqn. 6.28 is correct for $t-1$, then:

- **1.** By local pull-back inequality (Eqn. 5.3.1) and the bound of current estimation $\hat{\mathbf{p}}^t$ (Eqn. 6.27), we have:

$$\begin{aligned} \|I^t(R_{j0}) - I_{\delta\mathbf{p}^{t-1}}(R_{j0})\| &= \|H(I_{\text{test}}, \hat{\mathbf{p}}^{t-1})(R_{j0}) - I_{\delta\mathbf{p}^{t-1}}\| & (6.29)\\ &= \|I_{\text{test}}(R_j(\hat{\mathbf{p}}^{t-1})) - I_{\delta\mathbf{p}^{t-1}}\| \leq \eta_j r_t & (6.30) \end{aligned}$$

- **2.** Since Eqn. 6.30 together with the inductive hypothesis $\|\delta\mathbf{p}^{t-1}\|_\infty \leq r_t$ satisfies the

premise of Thm. 6.2.1, a prediction $\tilde{\mathbf{p}}_j(S_j)$ from j-th patch yields:

$$\|\tilde{\mathbf{p}}_j(S_j) - \delta\mathbf{p}^{t-1}(S_j)\|_\infty \leq \gamma_j r_t \tag{6.31}$$

- **3.** Eqn. 6.31 means for every landmark k:

$$\|\tilde{\mathbf{p}}_j(k) - \delta\mathbf{p}^{t-1}(k)\|_\infty \leq \gamma_j r_t \quad \forall k \in S_j \tag{6.32}$$

Then for every landmark k, the averaged prediction $\tilde{\mathbf{p}}(k)$ over overlapping patch set $S[k] = \{j : j \in [t], k \in S_j\}$ can also be bounded:

$$\begin{aligned} \|\tilde{\mathbf{p}}(k) - \delta\mathbf{p}^{t-1}(k)\| &= \left\| \frac{1}{\#S[k]} \sum_{j \in S[k]} \hat{\mathbf{p}}_j(k) - \delta\mathbf{p}^{t-1}(k) \right\| && (6.33) \\ &\leq \frac{1}{\#S[k]} \sum_{j \in S[k]} \|\hat{\mathbf{p}}_j(k) - \delta\mathbf{p}^{t-1}(k)\|_\infty && (6.34) \\ &\leq \frac{1}{\#S[k]} \sum_{j \in S[k]} \gamma_j r_t \leq \bar{\gamma} r_t && (6.35) \end{aligned}$$

- **4.** Finally, the residue $\delta\mathbf{p}^t$ after adding prediction $\tilde{\mathbf{p}}$ of layer t satisfy:

$$\|\delta\mathbf{p}^t\| = \|\delta\mathbf{p}^{t-1} - \tilde{\mathbf{p}}\| \leq \bar{\gamma} r_t = r_{t+1} = \bar{\gamma}^t r_1 \tag{6.36}$$

Theorem 6.3.2 (The Number of Samples Needed) *The total number N of samples needed is bounded by:*

$$N \leq C_3 C_1^d + C_2 \log_{1/\bar{\gamma}} 1/\epsilon \tag{6.37}$$

where $C_1 = 1/\min_j \alpha_j$, $C_2 = 2^{1/(1-\bar{\gamma}^2)}$ and $C_3 = 2 + c_{SS}(\lceil \frac{1}{2} \log_{1/\bar{\gamma}} 2K/d \rceil + 1)$.

Proof We divide our analysis into two cases: $d = 2K$ and $d < 2K$, where K is the number of landmarks. $d > 2K$ is not possible.

Case 1: $d = 2K$

First let us consider the case that the intrinsic dimensionality of deformation field d is just $2K$. Then the root dimensionality $d_1 = 2K$ (twice the number of landmarks). By Assumption 5.2.1, the dimensionality d_t for layer t is:

$$d_t = \beta r_t^2 = \frac{d_1}{r_1^2} r_t^2 = \bar{\gamma}^{2t-2} d_1 \tag{6.38}$$

Any patch $j \in [t]$ has the same degrees of freedom since by Assumption 5.2.1, d_j only depends

on r_j, which is constant over layer t.

For any patch $j \in [t]$, we use at most N_j training samples:

$$N_j \leq \left(\frac{1}{\alpha_j}\right)^{d_t} \tag{6.39}$$

to ensure the contracting factor is indeed at least $\gamma_j \leq \bar{\gamma}$. Note for patch j, we only need the content within the region R_{j0} as the training samples. Therefore, training samples of different patches in this layer can be stitched together, yielding samples that cover the entire image. For this reason, the number N_t of training samples required for the layer t is:

$$N_t \leq \arg\max_{j\in[t]} N_j \leq C_1^{d_t} = C_1^{\bar{\gamma}^{2t-2}d_1} \tag{6.40}$$

for $C_1 = 1/\min_j \alpha_j$. Denote $n_t = C_1^{\bar{\gamma}^{2t-2}d_1}$. Then we have:

$$N \leq \sum_{t=1}^{T} N_t \leq \sum_{t=1}^{T} n_t \tag{6.41}$$

To bound this, just cut the summation into half. Given $l > 1$, set T_0 so that

$$\frac{n_{T_0}}{n_{T_0+1}} = n_{T_0}^{1-\bar{\gamma}^2} \geq l, \quad \frac{n_{T_0+1}}{n_{T_0+2}} = n_{T_0+1}^{1-\bar{\gamma}^2} \leq l \tag{6.42}$$

Thus we have

$$\sum_{t=1}^{T} n_t = \sum_{t=1}^{T_0} n_t + \sum_{t=T_0+1}^{T} n_t \tag{6.43}$$

The first summation is bounded by a geometric series. Thus we have

$$\sum_{t=1}^{T_0} n_t \leq C_1^{d_1} \sum_{t=1}^{T_0} \left(\frac{1}{l}\right)^{t-1} \leq \frac{C_1^{d_1}}{1-1/l} = \frac{l}{l-1} C_1^{d_1} \tag{6.44}$$

On the other hand, each item of the second summation is less than $l^{1/(1-\bar{\gamma}^2)}$. Thus we have:

$$\sum_{t=T_0+1}^{T} n_t \leq l^{1/(1-\bar{\gamma}^2)} T \tag{6.45}$$

Combining the two, we then have:

$$N \leq \frac{l}{l-1} C_1^{d_1} + l^{\frac{1}{1-\bar{\gamma}^2}} T \tag{6.46}$$

for $T = \lceil \log_{1/\bar{\gamma}} 1/\epsilon \rceil$. Note this bound holds for any l, e.g. 2. In this case, we have

$$N \leq 2C_1^{d_1} + C_2 T \tag{6.47}$$

for $C_2 = 2^{\frac{1}{1-\bar{\gamma}^2}}$.

Case 2: $d < 2K$

In this case, setting $d_1 = 2K$, finding T_1 so that $d_{T_1} \geq d$ but $d_{T_1+1} < d$ in Eqn. 6.38, yielding:

$$T_1 = \left\lceil \frac{1}{2} \log_{1/\bar{\gamma}} 2K/d \right\rceil + 1 \tag{6.48}$$

Then, by Assumption 5.2.1, from layer 1 to layer T_1, their dimensionality is at most d. For any layer between 1 and T_1, N_t is bounded by a constant number:

$$N_t \leq c_{SS} C_1^d \tag{6.49}$$

The analysis of the layers from T_1 to T follow case 1, except that we have d as the starting dimension rather than $2K$. Thus, from Eqn. 6.47, the total number of samples needed is:

$$N \leq (T_1 c_{SS} + 2) C_1^d + C_2 T \tag{6.50}$$

∎

6.4 Empirical Upper Bounds For Images

Using the relaxed Lipschitz condition (Eqn. 6.3 and Eqn. 6.4), we are able to predict which patch are hard and which are easy for deformation estimation, by analyzing the monotonous curve $\gamma = \gamma(\alpha)$. For this, we set the contraction factor $\gamma = 0.95$ and compute the largest $\alpha_{0.95} = \gamma^{-1}(0.95)$, whose inverse gives the lowest sample complexity per dimension.

To empirically estimate the curve $\gamma = \gamma(\alpha)$, we randomly generate 1000 pairs of $(\mathbf{p}, I_{\mathbf{p}})$ and compute $M = 499500$ pairs of image differences $\{\Delta I_m\}$ and corresponding parameter differences $\{\Delta \mathbf{p}_m\}$ from a template image with 2D translation and in-plane rotation up to $\pm\pi/8$ ($d = 3$ dimensions). Then Alg. 4 is used to efficiently estimate the curve in $O(M \log M)$.

Fig. 6.4 shows sample complexity per dimension ($1/\alpha_{0.95}$) for exemplar images. Note that images with a salient object and uniform background only need a few samples per dimension, while images with repetitive patterns require more samples.

According to Theorem 6.2.1, for easy images, $1/\alpha_{0.95} \approx 5$, and for deformation that contains

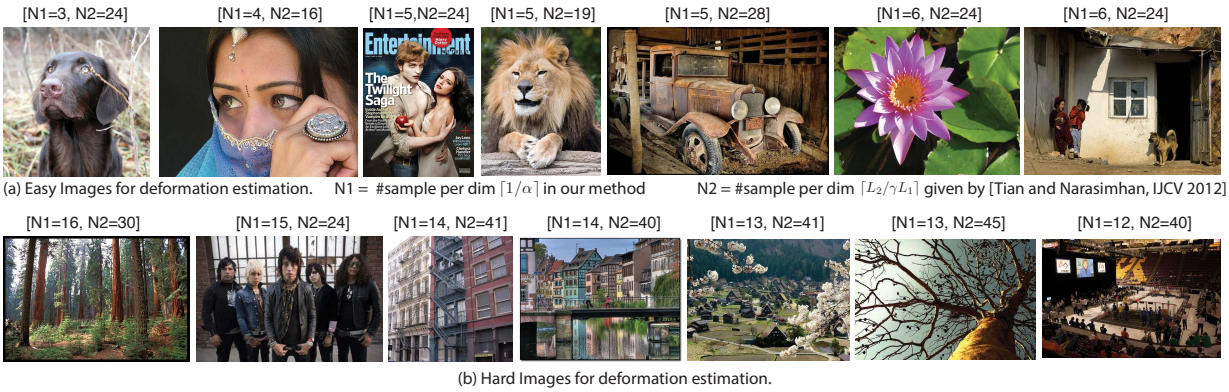


Figure 6.4: Exemplar images and the theoretical bounds for the number of samples needed per dimension. For each bracket, the first number is our bound (given by $1/\alpha_{0.95}$), while the second number from Data-Driven Descent (given by $L_2/\gamma L_1$ with $\gamma = 0.95$). **Top Row:** Images with a salient object and clean background require only a few samples per dimension. **Bottom Row:** Images with repetitive patterns require more samples per dimension. In both cases, our bound is smaller than that given by Data-Driven Descent.

only translation and rotation, $k = 4$ and $c = 2 + \sqrt{2}$ (See Sec. 13.2) and the number of samples needed is $(5 \cdot (2 + \sqrt{2})^4 = 84926$, while for hard images, $1/\alpha_{0.95} \approx 12$ and the number of samples needed is $(12 \cdot (2 + \sqrt{2}))^4 = 2817654$. Although this number may be more than practically necessary, it gives a sense of difficulty levels of images. In contrast, the number of samples needed ($L_2/\gamma L_1$) per dimension suggested in Data-driven Descent gives a much looser bound.

6.5 Experiments on Synthetic Data

We now show our algorithm works well for synthetic data. For all the experiments, our approach adopts a hierarchical structure using a grid of 256 landmarks with $\bar{\gamma} = 0.7$ and $T = 8$ layers. We use Thin-Plate Spline [10] as the bases function with proper normalization. While our theory gives an upper bound on the sample complexity, practically we found 350 training samples over all layers suffice for good performance.

6.5.1 Convergence Behavior

We artificially distorted 100 images with a 20-dimensional global warping field specified in Eqn. 3.5. For each image, its 10 distorted versions are generated with random parameters, which are estimated using Data-driven Descent and using Top-Down Hierarchical Prediction.

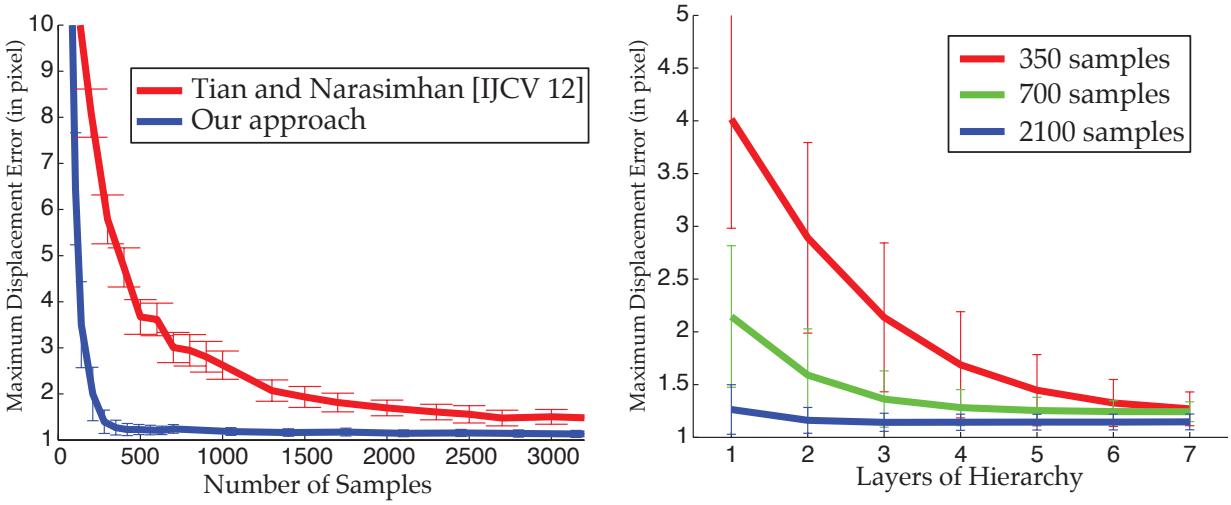


Figure 6.5: Performance of the proposed algorithm. **Left:** Performance comparison with Data-Driven Descent. Accuracy of Top-Down Prediction improves much faster than Data-Driven Descent with the same number of samples. To obtain the same level of accuracy, Top-Down Hierarchy needs 400 samples, while Data-Driven Descent requires 10000 samples or more. Our approach also has lower variance in performance. **Right:** Convergence behavior of our approach with different number of training samples.

Fig. 6.5 shows the performance comparison. Our algorithm obtains much better performance and lower variance compared to TN with the same number of training samples. Note that the strong drop in error shows that our method achieves very high accuracy by adding very few samples once it starts to work. This coincides with the theoretical analysis, which says the number of training samples needed per layer decays doubly exponentially with more layers.

6.5.2 Deformation Estimation on Repetitive Patterns

We further test our approach on synthetic data containing distorted repetitive patterns, and compare it with previous methods. From an undistorted template (240-by-240), we generate a dataset of 200 distorted images, each with labeled 49 points. The deformation field is created by random Gaussian noise without temporal continuity.

The overall degree of freedom for this dataset is very high (50 dimensions are needed to achieve < 1 pixel reconstruction error). It is in general impossible to have sufficient number of samples for global optimality conditions to be satisfied. However, practically our method still works well.

We compare our approach to the following previous methods: Lucas-Kanade (LK) [6], Data-driven Descent (TN), Free-form registration (FF) [69], Explicit Shape Regression (ESR) [14]

	LK	TN	ESR	FF	SR	Ours
RMS	14.79	6.44	8.98	7.29	98.94	**5.63**
sec/frame	11	77	**0.012**	35	1.25	0.10

Table 6.1: Performance comparison of different approaches, including Lucas-Kanade (LK) [6], Data-driven Descent (TN) (Chapter 4), Free-form registration (FF) [69], Explicit Shape Regression [14] and SIFT matching with outlier removal using RANSAC (SR) [53]. Ours is the best performer and second best in time cost per frame.

#Training	1	10	20	50	100	300
ESR	8.98	8.86	7.88	5.90	4.24	2.75
Ours	5.63	5.61	5.35	5.11	4.78	4.32

Table 6.2: Performance between our approach and ESR [14] with more training samples. ESR requires more training samples to work well.

and SIFT matching with outlier removal using RANSAC (SR) [53]. LK and TN use a local parametric deformation model. LK uses local affine bases of size 100-by-100, and TN uses a 20-dimensional smooth bases of size 57-by-40 (See Sec. 4.6.1). LK, FF and TN compute dense deformations and Top-Down Hierarchy outputs 256 predicted landmarks, from which 49 landmark locations are interpolated. The KLT tracker [83] requires temporal information and will be compared in the real video sequence.

For one image, the RMS error is computed between the estimated landmark locations $\hat{p}$ and ground truth locations $\mathbf{p}$ as $RMS = \sqrt{\frac{1}{K}\sum_{i=1}^{K} \|\mathbf{p}(i) - \hat{\mathbf{p}}(i)\|^2}$. For multiple images, averaged RMS is reported.

Table 6.1 compares the performance. Due to repetitive patterns, previous approaches fail to estimate the landmarks correctly. SIFT matching fails completely. The prediction of ESR is restricted to be on the linear shape subspace spanned by the training samples. Thus, it is insufficient to use the template to capture the subspace of a complex deformation field. LK and FF are stuck in local maxima despite their coarse-to-fine implementations. Our approach obtains the best performance. Fig. 6.6 shows the progression of our algorithm. In terms of speed, our approach is second only to ESR, which uses a fast boosting framework.

Influence of multiple layers. It is interesting to see how the performance changes if we switch off the first L layers of predictors. As shown in Table 6.3, the first two layers have less contribution on the performance than the rest of the layers. On the other hand, the lower 6 layers indeed help the performance. Fig. 6.7 demonstrates how prediction from coarse layers (large patch) help the lower layer (small patch) find correct correspondences in repetitive patterns, justifying the hierarchy.

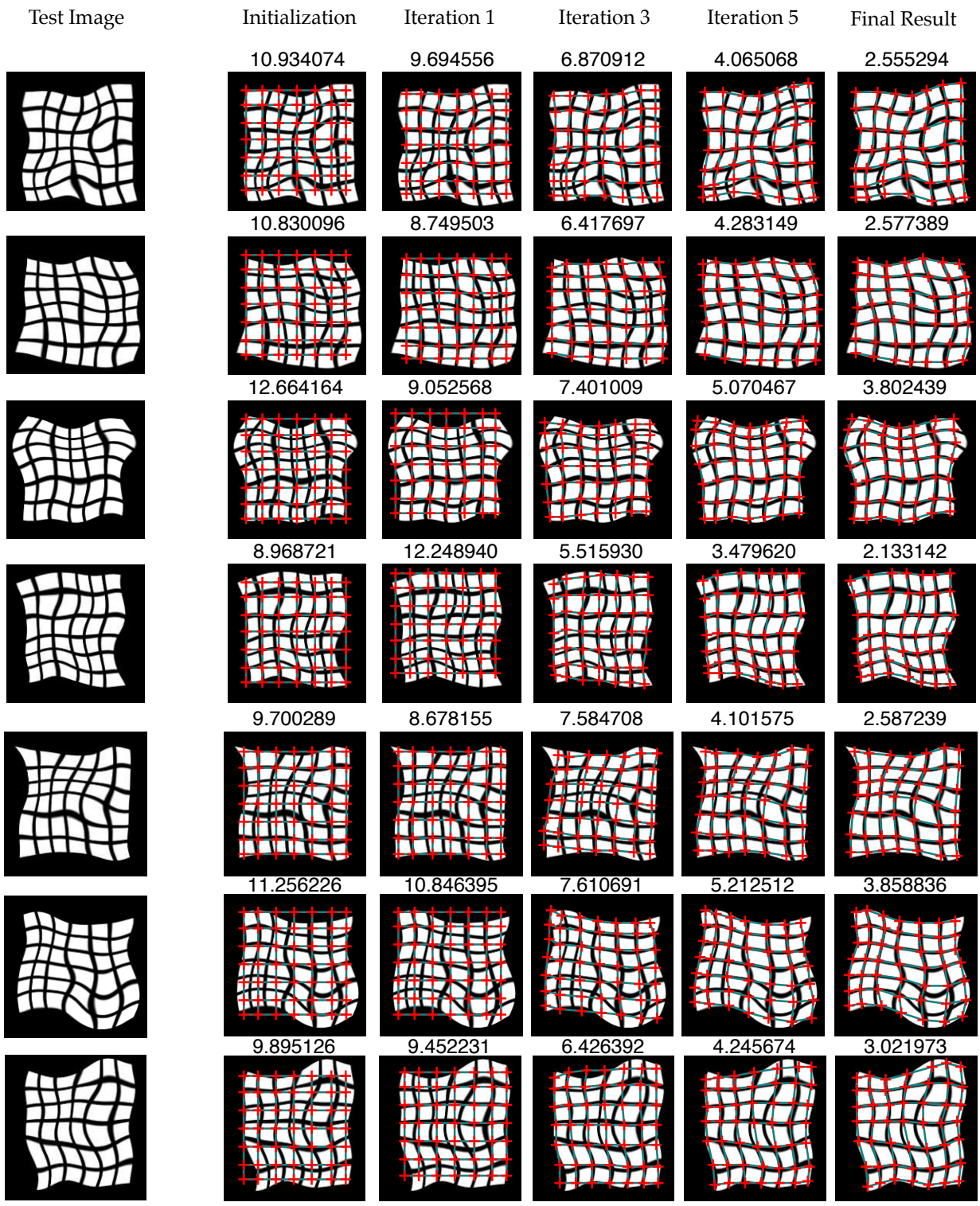


Figure 6.6: Demonstration of the iterative procedure of our algorithm. Starting from initialization, the algorithm applies predictors of different layers to estimate the landmark locations. Numbers on top show RMS errors.

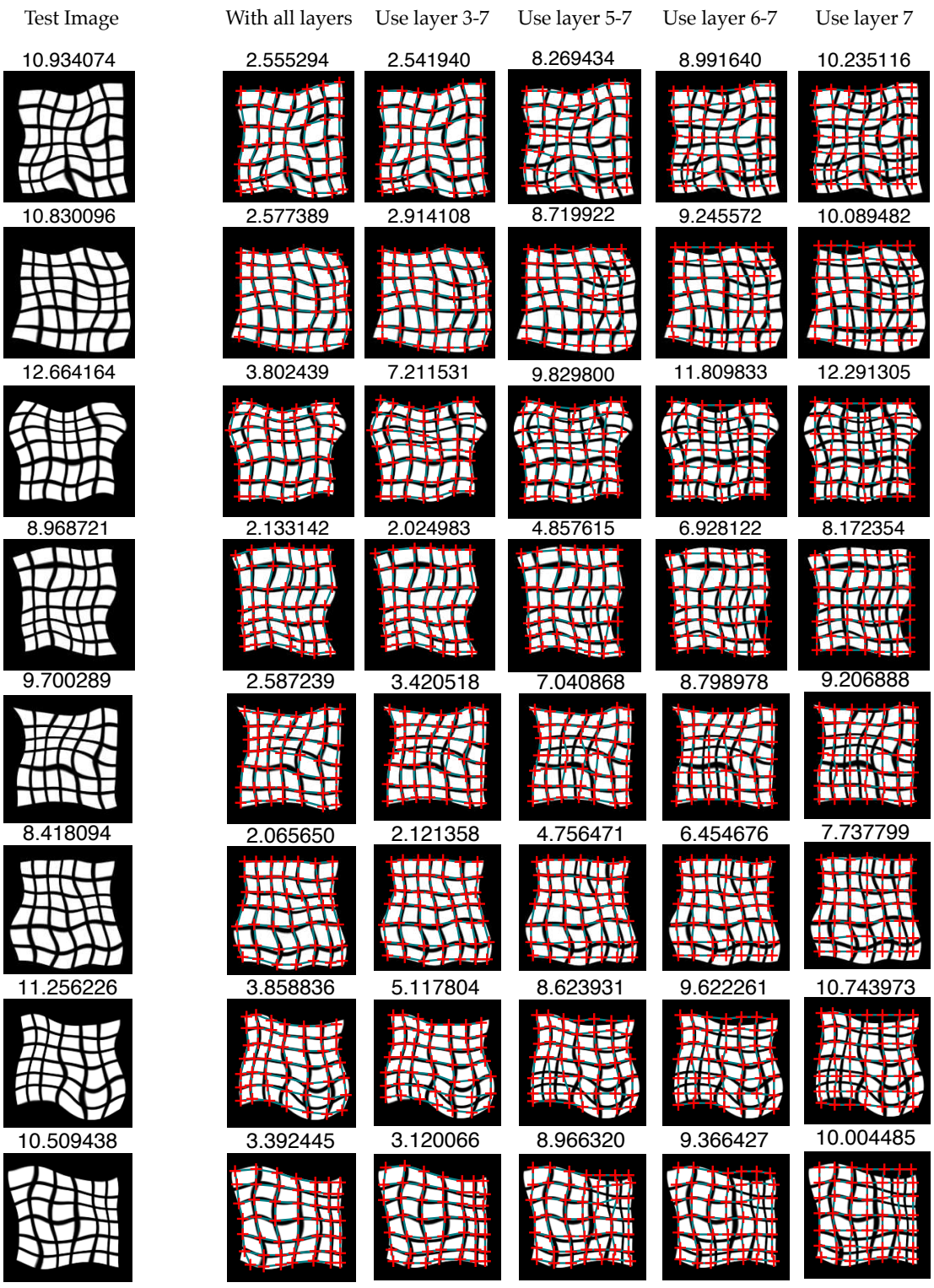


Figure 6.7: Performance changes if the first L layers are switched off. When more layers are switched off, the algorithm is unable to identify global deformation and performs essentially the same as local template matching at each landmark. Layer 3-4 is critical for getting a good estimation of the landmarks on the synthetic data.

L	0	1	2	3	4	5	6
RMS	5.63	5.20	5.14	5.83	6.72	7.95	8.74

Table 6.3: Performance on synthetic data if the first L layer of predictors are switched off, showing the bottom layers play a critical role for performance.

6.6 Real Experiments

We also apply our framework to real world scenarios such as water distortion, cloth deformation and registration of medical images. In Fig. 6.8, contour tracking is achieved by interpolating contour points from frame correspondences, while the contour of the first frame is manually labeled. In Fig. 6.9, tracked mesh is shown.

The three water distortion sequences (Row 1-2 in Fig. 6.8, Row 1 in Fig. 6.9) and one cloth sequence (Row 3 in Fig. 6.8) are in Chapter 4). Two cloth sequences (Row 2-3 in Fig. 6.9) are from [95] and [58]. The medical sequence of cardiac magnetic resonance images (4th row in Fig. 6.8) is from [121]. We captured the cloth sequence in the 5th row of Fig. 6.8.

For the sequences on the 4th row of Fig. 6.8 and the 1st row of Fig. 6.9, we use temporal information by adding training samples generated from perturbing the final estimation of the previous frame. This slows down the processing to 0.3-0.5fps, yet is still faster than previous approaches. For other sequences, our algorithm runs at around 3-4 fps.

Note that our method successfully estimates the deformations. In comparison, SIFT+RANSAC only obtains a sparse set of distinctive matches, not enough for estimating a nonrigid deformation (even if we are using Thin-Plate Spline). TN can capture detailed local deformations but not global shifts of the cloth without modeling the relationship between local patches. KLT trackers lose the target quickly and localize contour inaccurately.

We also quantitatively measure the landmark localization error using the densely labeled dataset provided in [100], which contains 30 labeled frames, each with 232 landmarks. In terms of RMS, LK gives 5.20, FF gives 3.93, TN gives 2.51 while our approach gives 3.29. Our framework is only second to TN, which is much slower.

We have tested our algorithm on existing datasets of deformable objects proposed by [72, 73]. Although no groundtruth is available, our performance is close to their published results (e.g. 4.10 mean pixel distance difference in cushion video [72] and 4.43 in bed-sheet video [73]). All video sequences are 404-by-504.

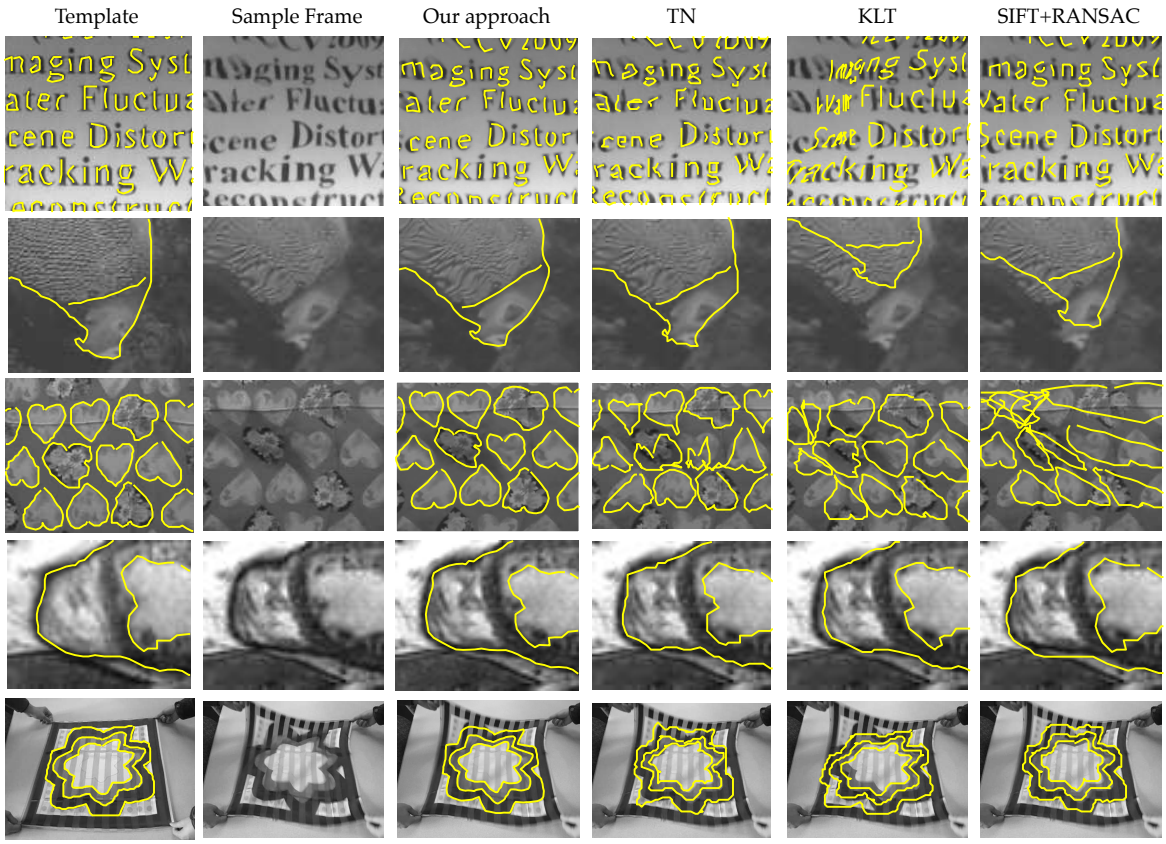


Figure 6.8: Example contour localization results given by our approach, TN [100], KLT [83], and SIFT matching with RANSAC [53]. Each row is a video sequence, two from underwater imaging, two from cloth deformation and the final one is from medical imaging. For each dataset, one sample frame is shown. The contours are drawn manually for the template image (1st column), and are transferred to every video frame after the correspondence was found. Our approach is stable and better than other approaches. (This figure is best viewed in color)

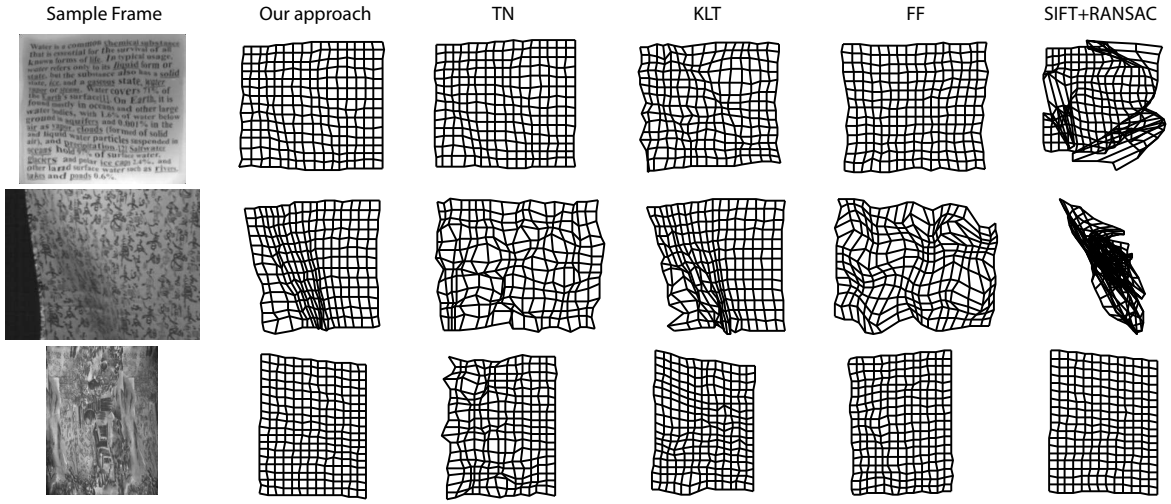


Figure 6.9: Example dense correspondence results given by our approach, TN, KLT, FF and SIFT matching with RANSAC. Each row is a video, two from cloth deformation and one from underwater imaging. The mesh is a regular grid on the template.

Chapter 7

Bottom-Up Hierarchical Prediction: Breaking the Curse of Dimensionality

The Top-down Hierarchical Prediction proposed in Chapter 6 improves the sample complexity from $O(C^d \log 1/\epsilon)$ to $O(C_1^d + C_2 \log 1/\epsilon)$, further decoupling the dimensionality d of deformation with the accuracy $1/\epsilon$.

However, the curse of dimensionality (C_1^d) still remains. The major reason is that in the Data-driven Descent and the topmost layer of Top-down Hierarchical Prediction, a predictor is required to take care of all the degrees of freedom (DoF), although the precision in each DoF can be very rough. As a result, to obtain any guaranteed predictions with Nearest Neighbor on the top-most layer, the exponential factor of C_1^d is inevitable.

To avoid the exponential factor C_1^d, a natural solution is to *discard* some degrees of freedom on each layer when following the upstream of hierarchy. This is done by building *invariant representations* that are insensitive to discarded degrees of freedom in a bottom-up manner. Then the top-most layer only sees a few degrees of freedom and the computation is tractable.

Then the problem becomes how much DoFs should be discarded in each layer. Obviously, discarding too many degrees of freedom is also a bad idea since the top levels would have insufficient information to decide on. The basic workflow for both Data-driven Descent and Top-down Hierarchy is that the estimations given by the first few iterations (or the top levels), are guaranteed to be rough but correct, which can be used for further refinement. However, if the top levels are fed with insufficient information, then such a guarantee cannot be made.

Therefore, we have to choose to discard *some particular* degrees of freedom, in this thesis local deformation, so that the top layers are still able to make an *unambiguous* decision, in this thesis to estimate the global deformation, while the computational burden is manageable. The minimal requirement is: any decision made in the top level, when propagated downwards, should

solve any unsettled ambiguities in the lower layers. This determines which degrees of freedom should be discarded in each layer of the hierarchy, and which shall be kept and handed to the higher layer.

Interestingly, it turns out that such an intuition can again be measured quantitatively by (a variant of) relaxed Lipschitz conditions (Eqn. 6.3 and Eqn. 6.4). Since the representations are computed in a bottom-up manner, it can be proven that if all the children satisfy the conditions, so does their parents with different Lipschitz constants. By induction, the Lipschitz conditions can be propagated upwards until the top-most level. Therefore, they only need to be assumed true in the lowest level. Compared to the Top-Down case in which all patches are assumed to satisfy the conditions, Bottom-Up Model assumes far less, not to mention the stricter one used in Data-driven Descent Eqn. 3.22.

This chapter follows the thoughts and proposes Bottom-up Hierarchical Prediction for deformation estimation. This algorithm is a two-pass procedure. First, from bottom to top, the image evidence is transferred upwards to build invariant representations. Once the decision is made in the top-most level, it is propagated back to the lower level to help resolve any local ambiguities. As a result, the sample complexity is reduced to $O((C/\epsilon)^{d_0})$ for $d_0 \ll d$, essentially breaking the curse of dimensionality.

7.1 Intuition

As usual, let us start with a story to explain the intuition. Imagine Robert was a Baron ruling a land. When the land is only a few square kilometers living with only a handful number of residents, our diligent Robert can handle all the daily affairs happening on the land by himself (Fig. 7.1(a)). All residents are happy since their ruler knows them well, and Robert is happy since he knows all details of the land, including how crops grow in each season, the amount of livestock each resident has and even the color of houses. Knowing details enables Robert to fulfill his duty, i.e., to make decision on specific laws, and report annual gross incomes of the land to the king.

Time flies and Robert is now promoted to be Duke. He now governs a larger land with more residents. The daily affairs now become much more overwhelming to him, even he sleeps only 4 hours a day, as shown in Fig. 7.1(b). One day he finally realizes that he cannot manage it in full details. Thus, Robert asks some close friends, Edward, Terry and Peter to help him. All friends now have a Baron title and rules one part of the land, and reports to Robert (See Fig. 7.1(c)).

Then problem arises. As a novice, Edward does not know what to report. Robert thus tells him: “Easy, just report the annual gross income so that I can add up your numbers.” As for the de-

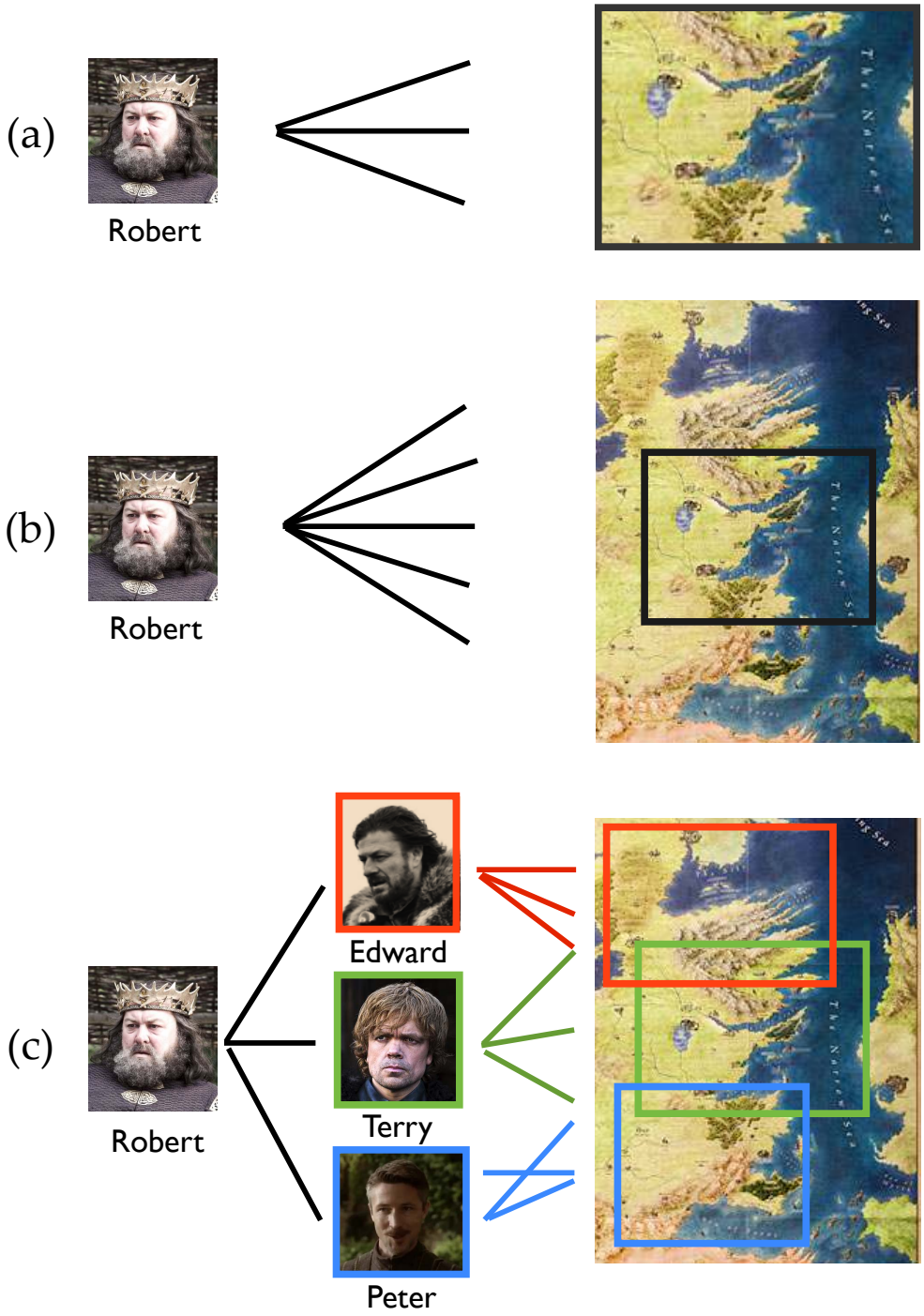


Figure 7.1: The story of Robert ruling a land. **(a)** When the land is small, Robert, as a baron, can take care of all the details of his land. **(b)** Robert now becomes a duke ruling a larger land. The increased details overwhelm Robert even he works 20 hours a day. **(c)** Robert asked his three friends, each taking care of part of his land. The friends report to him only the important information and take care of the details by themselves.

tails such as the color of the houses, his friends may not report it. Sometimes Terry cannot make a decision due to lack of overview information, then Robert asks them to report any evidences to him, let him decide the general principles, and send the principles back to help decision.

In this story, our diligent Robert used to be a Nearest Neighbor predictor that takes care of every details for a prediction. But the *overwhelming details of managing a land*, i.e., the increasing (exponential) complexity, is beyond his ability. Therefore, he uses a hierarchical structure. Each friend discards details (e.g., *the color of the houses*), and only report supportive information (e.g., *annual income*) that makes Robert's decision unambiguous. Finally, Robert gives them the general principles that help them decide affairs in their land, which is the top-down propagation procedure.

7.2 Hierarchical Decomposition of Deformations

7.2.1 Principle of Lossy Decomposition

We follow Sec. 5.4 to build the hierarchical structure. In the Top-down Hierarchical Predictions, the degrees of freedom of each vertex V_j is represented by the local parameters $\mathbf{p}(S_j)$, where S_j is the subset of dominant landmarks. Since $\mathbf{p}(S_j)$ is a $|S_j|$-by-2 matrix, the possible number of states of $\mathbf{p}(S_j)$, is small for small patches, but could be unmanageable for large ones.

To solve this problem, we need to reduce the number of states for $\mathbf{p}(S_j)$. Denote h_j as the *held state* and g_j as the *discarded state* for vertex V_j. The state space $\mathcal{H}_j$ contains all possible choices of h_j, while $\mathcal{G}_j$ contains all possible choices of g_j. Conceptually, $\mathbf{p}(S_j)$ can be decomposed into the following three components:

$$\mathbf{p}(S_j) = \underbrace{h_j}_{\text{DoF held for upstream}} \otimes \underbrace{g_j}_{\text{DoF discarded}} \otimes \underbrace{\bigotimes_{k \in ch(j)} \mathbf{p}(S_k)}_{\text{DoF in children}} \tag{7.1}$$

Intuitively, h_j is the coarse state of patch j that may be useful in the top-level prediction. For example, where is the center location of the patch, how much it rotates and how it scales compared to its counterpart in the template I_0. h_j only contains a few degrees of freedom. g_j is the state that is also coarse but is unrelated with the top-level decision and can be discarded. Finally, $\bigotimes_{k \in ch(j)} \mathbf{p}(S_k)$ is the finer state that is encoded in the lower layers.

Mathematically, using our favorite deformation model (Eqn. 5.2):

$$W(\mathbf{x}; \mathbf{p}) = \mathbf{x} + B(\mathbf{x})\mathbf{p} \tag{7.2}$$

that may contain both global and local deformation, Eqn. 7.1 can be made more specific. As shown in Fig. 7.2, the total degrees of freedom, or dimensionality in Eqn. 7.2, or the rank of $B(\mathbf{x})$ is around $\min(D, 2K)$, while D is the number of bases and K is the number of landmarks. Note the bases may not be orthogonal and the intrinsic dimensionality d (which is precisely the rank of $B(\mathbf{x})$) may be smaller than D. However, unless in very specific cases, $\min(D, 2K)$ can be a good estimate of d.

$\min(D, 2K)$ is in general very large, and a simple nearest neighbor on this space is intractable. However, if we only consider a local region R_{j0} and its dominant landmarks S_j, the rank of submatrix $B(R_{j0})$ within R_{j0} is $\min(D, 2|S_j|)$ and is low. As a result, the deformation within R_{j0} can be handled (Fig. 7.2(a)). Note the deformation within R_{j0} could be contributed by both the global and local deformation. However, within this small region, their contribution *collapses*.

For a parent j, its total DoF, aggregated from all of its children $k \in ch(j)$, is definitely larger (Fig. 7.2(b-c)). Without throwing any DoFs, the root vertex will again has DoF of size $\min(D, 2K)$ and is not manageable. Therefore, the total DoF is decomposed into two components, h_j and g_j. h_j is the retained components that will help for high-level decision, in particular, for determining the global components of deformation, while g_j is the discarded components related to only local deformation (Fig. 7.2(d)). Note since the contributions from global and local deformation are largely indistinguishable within the small region, most of the information gathered from R_{j0} may be related to high-level decision and the discarded components g_j may be "small". However, when we follow the hierarchy to the top level, when supports of patches become larger and larger, the more and more detailed information will be discarded. As a result, for every patch of the entire hierarchy, its information to be processed is manageable.

7.2.2 Hierarchical Deformable Mixture Model (HDMM): A Concrete Example

From the analysis of the previous section, there has to be a relationship in terms of states (h, g) between a parent vertex V_j and its child V_k for $k \in ch(j)$. Some intuitions might help in building such a relationship between (h_j, g_j) and (h_k, g_k). For example, both the parent part and the child parts are located nearby (i.e., $\mathbf{u}_j$ and $\mathbf{u}_k$ is close to each other), both are under 30 degree rotation (i.e., The mixture type variables z_j and z_k are closely related), etc.

As a concrete example, we represent the somehow abstract held state h_j as $(\mathbf{u}_j, z_j)$, where $\mathbf{u}_j$ is the 2D location of patch j and $z_j \in \mathcal{Z}_j$ is the *type* ID of the patch. The type variable z_j accounts for any variations that are not 2D translation, such as rotation, scaling, location deformation and

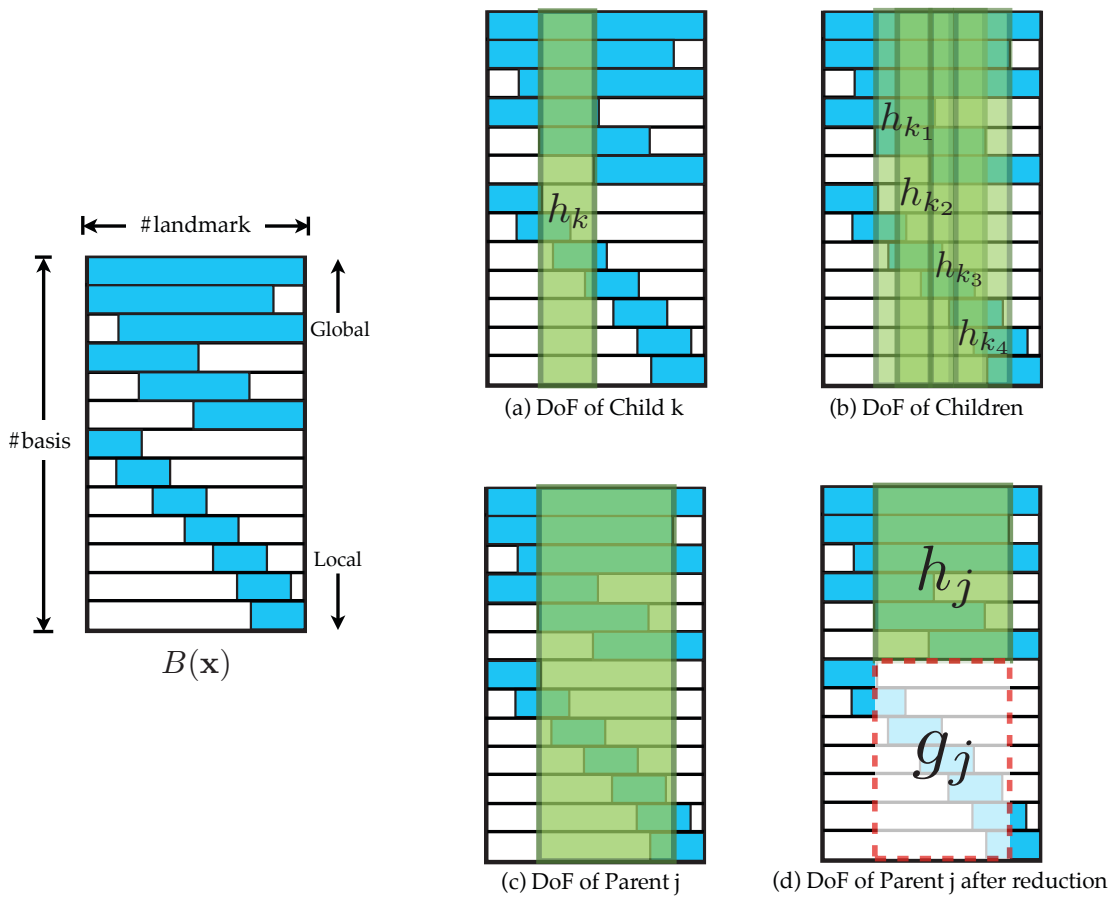


Figure 7.2: Decomposition of bases $B(\mathbf{x})$ into a bottom-up hierarchical model. **Left:** The bases $B(\mathbf{x})$ that encodes both global and local deformations. Each column of $B(\mathbf{x})$ is a basis while each row represents a pixel/landmark. **Right:** *(a)* a child vertex k that only covers a small subset of landmarks and its DoF (illustrated by a green shaded area) is limited. *(b)* combining the children together, their DoFs also accumulate. *(c)* The total DoF of a parent vertex j is the total DoFs of all its children. *(d)* The parent vertex only keep important DoFs (h_j) and discard details g_j.

other appearance changes (e.g., open/close palm, upright/horizontal arm in human part modeling, occluded/unoccluded regions in cloth modeling). Statistically, z_j models the appearance variation of patch j as a appearance mixture.

Mathematically, we define the *compatibility relation*, denoted as $(h_j, g_j) \sim h_k$ as follows:

Definition 7.2.1 *We define $(h_j, g_j) \sim h_k$ if and only if the following condition holds:*

$$\mathbf{u}_k = \mathbf{u}_j + \delta\mathbf{u}_{jk}(z_j) + g_{jk} \tag{7.3}$$

$$z_k \in \psi^+_{jk}(z_j) \tag{7.4}$$

In Eqn. 7.3, there are two terms that need explanation. Firstly, $\delta\mathbf{u}_{jk}(\cdot)$ is the *relative shift* from parent j to child k indexed by parent type z_j. Intuitively, for different type z_j of j-th part, the spatial arrangements of its children could be different. The discarded state g_j is a $|ch(j)|$-by-2 matrix, with each row corresponding to a local displacement of a child. Intuitively, this is the "error" between the predicted location from parent

$$\mathbf{u}_{jk}(z_j) \equiv \mathbf{u}_j + \delta\mathbf{u}_{jk}(z_j)$$

to the actual location of child $\mathbf{u}_k$. Furthermore, we require such error to be bounded:

$$\|g_j\|_\infty = \max_{k \in ch(j)} \|g_{jk}\|_\infty \leq \frac{\bar{g}_j}{2} \tag{7.5}$$

Second, Eqn. 7.4 is the *compatibility term* between parent type z_j and child type z_k. For example, in human pose modeling, for part j being the entire arm and k being the lower arm, if $z_j = upright$, then z_k cannot be horizontal. Therefore, given z_j, there is a set $\psi^+_{jk}(z_j)$ that contains allowable z_ks. Note that from Eqn. 7.4, there could be multiple z_ks. Moreover, without knowing g_j (since it is discarded), we could not even recover the precise location $\mathbf{u}_k$ of each child k. This is a desirable property for downstream operation since from children to parent, information loss is a must. As a dual operation to $\psi^+_{jk}(z_j)$, we define $\psi^-_j(z_{ch(j)})$ that returns compatible z_j with all $\{z_k\}_{k \in ch(j)}$:

$$\psi^-_j(z_{ch(j)}) = \bigcap_{k \in ch(j)} (\psi^+_{jk})^{-1}(z_k) \tag{7.6}$$

where $(\psi^+_{jk})^{-1}(z_k)$ returns allowable z_js for a given z_k. For brevity, we use $z_k \sim z_j$ to show z_k is compatible with z_j.

Conversely, if we know all the children's held states $h_{ch(j)} \equiv \{h_k\}_{k \in ch(j)}$, the state of their

parent can be *uniquely* determined. Such a relationship is defined as follows:

$$[\mathbf{u}_j, z_j] = \arg \min_{\mathbf{u}_j, z_j \in \psi_j^-(z_{ch(j)})} \|\mathbf{u}_{jk}(z_j) - \mathbf{u}_k\|^2 \tag{7.7}$$

$$g_{jk} = \mathbf{u}_k - \mathbf{u}_{jk}(z_j) \tag{7.8}$$

7.2.3 Properties of HDMM

Distinctiveness of Types. Two types z_j and z'_j have to be distinct:

$$\|\delta\mathbf{u}_j(z_j) - \delta\mathbf{u}_j(z'_j)\| = 0 \implies z_j = z'_j \tag{7.9}$$

otherwise we can just remove duplicate types.

Metric for type variables. As discrete variables, type variables in general are not equipped with distances. However, intuitively, a type that represents "30 degrees of rotation" is closer to a type that represents "31 degrees of rotation", compared to a type with 60 degrees of rotation. To represent such a distance metric, for two types z_j and z'_j, we measure their distance with the help of relative shifts $\delta\mathbf{u}_{jk}$:

$$\|\delta\mathbf{u}_j(z_j) - \delta\mathbf{u}_j(z'_j)\| \equiv \max_{k \in ch(j)} \|\delta\mathbf{u}_{jk}(z_j) - \delta\mathbf{u}_{jk}(z'_j)\|_\infty \tag{7.10}$$

We can prove Eqn. 7.10 is indeed a metric for type z_j since it is nonnegative and symmetric, and from Eqn. 7.9 it is zero if and only if $z_j = z'_j$. Finally, since

$$\|\delta\mathbf{u}_{jk}(z_j) - \delta\mathbf{u}_{jk}(z''_j)\|_\infty \leq \|\delta\mathbf{u}_{jk}(z_j) - \delta\mathbf{u}_{jk}(z'_j)\|_\infty + \|\delta\mathbf{u}_{jk}(z'_j) - \delta\mathbf{u}_{jk}(z''_j)\|_\infty \tag{7.11}$$

for any $k \in ch(j)$, we have

$$\|\delta\mathbf{u}_{jk}(z_j) - \delta\mathbf{u}_{jk}(z''_j)\|_\infty \leq \max_{k \in ch(j)} \|\delta\mathbf{u}_{jk}(z_j) - \delta\mathbf{u}_{jk}(z'_j)\|_\infty + \max_{k \in ch(j)} \|\delta\mathbf{u}_{jk}(z'_j) - \delta\mathbf{u}_{jk}(z''_j)\|_\infty \tag{7.12}$$

and thus

$$\|\delta\mathbf{u}_j(z_j) - \delta\mathbf{u}_j(z''_j)\| = \max_{k \in ch(j)} \|\delta\mathbf{u}_{jk}(z_j) - \delta\mathbf{u}_{jk}(z''_j)\|_\infty \tag{7.13}$$

$$\leq \max_{k \in ch(j)} \|\delta\mathbf{u}_{jk}(z_j) - \delta\mathbf{u}_{jk}(z'_j)\|_\infty \tag{7.14}$$

$$+ \max_{k \in ch(j)} \|\delta\mathbf{u}_{jk}(z'_j) - \delta\mathbf{u}_{jk}(z''_j)\|_\infty \tag{7.15}$$

$$= \|\delta\mathbf{u}_j(z_j) - \delta\mathbf{u}_j(z'_j)\| + \|\delta\mathbf{u}_j(z'_j) - \delta\mathbf{u}_j(z''_j)\| \tag{7.16}$$

Besides, for parent-child pair (j,k), we define the following monotonous increasing function $\delta\bar{\mathbf{u}}^+_{jk}(\cdot)$ and $\delta\bar{\mathbf{u}}^-_{jk}(\cdot)$:

$$\begin{aligned}
\delta\bar{\mathbf{u}}^-_{jk}(\lambda) &= \max \|\delta\mathbf{u}_j(z_j) - \delta\mathbf{u}_j(z'_j)\| && (7.17)\\
&\text{s.t.} \quad \|\delta\mathbf{u}_k(z_k) - \delta\mathbf{u}_k(z'_k)\| \le \lambda,\ \ z_k \in \psi^+_{jk}(z_j),\ z'_k \in \psi^+_{jk}(z'_j)\\
\delta\bar{\mathbf{u}}^+_{jk}(\lambda) &= \max \|\delta\mathbf{u}_k(z_k) - \delta\mathbf{u}_k(z'_k)\| && (7.18)\\
&\text{s.t.} \quad \|\delta\mathbf{u}_j(z_j) - \delta\mathbf{u}_j(z'_j)\| \le \lambda,\ \ z_k \in \psi^+_{jk}(z_j),\ z'_k \in \psi^+_{jk}(z'_j)
\end{aligned}$$

Intuitively, $\delta\bar{\mathbf{u}}^-_{jk}(\cdot)$ gives an upper bound on how large the variance of relative shifts of parent j will be, if the child k change their types up to a limit 2λ. Similarly, $\delta\bar{\mathbf{u}}^+_{jk}(\cdot)$ gives an upper bound on how large the variance of child k will be, if the parent j changes its type up to a limit 2λ. Both functions are monotonously increasing:

$$\delta\bar{\mathbf{u}}^+_{jk}(\lambda_1) \le \delta\bar{\mathbf{u}}^+_{jk}(\lambda_2), \quad \lambda_1 \le \lambda_2 \tag{7.19}$$

$$\delta\bar{\mathbf{u}}^-_{jk}(\lambda_1) \le \delta\bar{\mathbf{u}}^-_{jk}(\lambda_2), \quad \lambda_1 \le \lambda_2 \tag{7.20}$$

since the larger λ is, the loose the constraint is, and the higher the maximum value will be. Therefore, the following mapping $\phi^+_{jk} \equiv I + \delta\bar{\mathbf{u}}^+_{jk}$ and $\phi^-_{jk} \equiv I + 2\delta\bar{\mathbf{u}}^-_{jk}$:

$$\phi^+_{jk}(\lambda) \equiv \lambda + \delta\bar{\mathbf{u}}^+_{jk}(\lambda) \tag{7.21}$$

$$\phi^-_{jk}(\lambda) \equiv \lambda + 2\delta\bar{\mathbf{u}}^-_{jk}(\lambda) \tag{7.22}$$

always have an inverse.

7.2.4 The Expressive Power of HDMM

One interesting question is what kind of deformation can be represented by HDMM? It turns out that the model is very expressive. As a specific example, consider the following deformation field $\mathbf{p}(\cdot)$ defined on a T-by-T regular grid $\mathbf{l}_k = (x_m, y_n)$ (See Fig. 7.3):

Definition 7.2.2 (Locally Smooth Deformation Field (LS-DF)) *A deformation field is locally smooth if for any k_1 and k_2, we have:*

$$\|\mathbf{p}(\mathbf{l}_{k_1}) - \mathbf{p}(\mathbf{l}_{k_2})\|_\infty \le c_S \|\mathbf{l}_{k_1} - \mathbf{l}_{k_2}\|_\infty \tag{7.23}$$

where $\mathbf{l}_k$ are a set of regular grid points on a 2D plane. Without less of generality, for two grid

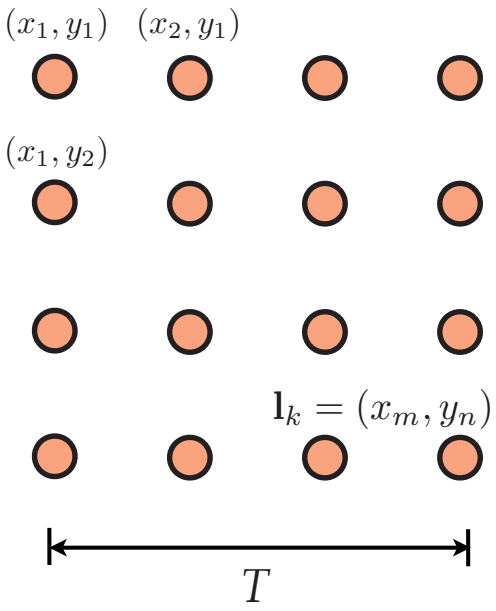


Figure 7.3: Regular grid used to construct Hierarchical Deformable Mixture Model.

points $\mathbf{l}_{k_1} = (x_{m_1}, y_{n_1})$ *and* $\mathbf{l}_{k_2} = (x_{m_2}, y_{n_2})$*, their distance is:*

$$\|\mathbf{l}_{k_1} - \mathbf{l}_{k_2}\|_\infty = \max(|m_1 - m_2|, |n_1 - n_2|) \tag{7.24}$$

The degrees of freedom for LS-DF is proportional with respect to the diameter of $\mathbf{l}_k$. Therefore, for grid points that cover a large range, to enumerate all possible such deformations is intractable.

However, one can construct a HDMM that has the same expressive power as LS-DF, and yet has a guaranteed solution with much lower computational complexity, which will be analyzed in great details in the next sections. The construction of such HDMM is as follows:

Definition 7.2.3 (HDMM for LS-DF) *We define a hierarchical structure with T layers. For any vertex j, we have:*

- $|\mathcal{Z}_j| = 1$. *That is, there is no type variable.*
- *No relative shifts:* $\delta\mathbf{u}_j \equiv 0$.
- $\bar{g}_j = 2c_S$

In layer t, there are t^2 vertices and each vertex V_{j_1,j_2} $(1 \le j_1, j_2 \le t)$ covers a sub-rectangle of landmarks S_{j_1,j_2}:

$$S_{(j_1,j_2)} = (x_{j_1}, x_{j_1+1}, \ldots, x_{j_1+T-t}) \times (y_{j_2}, y_{j_2+1}, \ldots, y_{j_2+T-t}) \tag{7.25}$$

Note that $|S_{(j_1,j_2)}| = (T-t+1)^2$. *From* $\{S_j\}$, *we thus can construct the hierarchical structure according to Sec. 5.4.*

The resulting HDMM looks like Fig. 5.5.

Then we are going to show that LS-DF can be represented by this HDMM. We start from the following lemma:

Lemma 7.2.4 (Properties of HDMM (Eqn. 7.2.3)) *The followings are true:*

- ***(a)*** *The pairwise distance* $\|\mathbf{l}_{k_1} - \mathbf{l}_{k_2}\|_\infty$ *is on a discrete set* $\{0, 1, \ldots, T-1\}$.
- ***(b)*** *For layer* t, *for any* $\mathbf{l}_{k_1}, \mathbf{l}_{k_2} \in S_{j_1,j_2}$, *we have*

$$\|\mathbf{l}_{k_1} - \mathbf{l}_{k_2}\|_\infty \leq T - t \tag{7.26}$$

- ***(c)*** *If* $\|\mathbf{l}_{k_1} - \mathbf{l}_{k_2}\|_\infty = l$, *then there exists a path* $P = [k_1, \ldots, j, \ldots, k_2]$ *that contains* $2l+1$ *vertices on the hierarchical structure from vertex* k_1 *to vertex* k_2. *Here* $t(j) = T - l$.

Proof Since $\mathbf{l}_k$ is a regular grid of size T-by-T, (a) is true. From the construction of $S_{(j_1,j_2)}$, (b) is true. Finally, for $\mathbf{l}_{k_1} = (x_{m_1}, y_{n_1})$, $\mathbf{l}_{k_2} = (x_{m_2}, y_{n_2})$, since:

$$\|\mathbf{l}_{k_1} - \mathbf{l}_{k_2}\|_\infty = \max(|m_1 - m_2|, |n_1 - n_2|) = l \tag{7.27}$$

at layer $T - l$ there exists a vertex j with $t(j) = T - l$ that covers both landmarks. From j to k_1 there exists a sequence of parent-child pair that contains l vertices (excluding j). Similarly there is a sequence of length l between j and k_2. Connecting them together yields the final path P.

and a definition:

Definition 7.2.5 *For any vertex* (j_1, j_2), *its grid point location is assigned to be:*

$$\mathbf{l}_{(j_1,j_2)} = (x_{j_1}, y_{j_2}) \tag{7.28}$$

Then we show the relationship between LS-DF and HDMM constructed from Definition 7.2.3.

Theorem 7.2.6 (HDMM accepts LS-DF) *If a deformation field defined on the regular grid* $\mathbf{l}_k$ *is LS-DF, then all the states* (h_j, g_j) *computed from Eqn. 7.7 and Eqn. 7.8 are valid:*

$$\|g_j\|_\infty \leq \frac{\bar{g}_j}{2} \tag{7.29}$$

Proof We first consider parent layer $T-1$ and child layer T. By Eqn. 7.26, for each parent vertex $j \in [T-1]$, we have for any $\mathbf{l}_{k_1}, \mathbf{l}_{k_2} \in S_j$ (or equivalently $k_1, k_2 \in ch(j)$), $\|\mathbf{l}_{k_1} - \mathbf{l}_{k_2}\|_\infty \leq 1$. By

Eqn. 7.23, we thus have

$$\|\mathbf{p}(\mathbf{l}_{k_1}) - \mathbf{p}(\mathbf{l}_{k_2})\|_\infty \leq c_S \tag{7.30}$$

Therefore, by setting $\mathbf{u}_j = \text{mean}_{k \in ch(j)} \mathbf{p}(\mathbf{l}_k)$, for any k, we have

$$\|g_{jk}\|_\infty = \|\mathbf{u}_j - \mathbf{p}(\mathbf{l}_k)\|_\infty \leq c_S = \frac{\bar{g}_j}{2} \tag{7.31}$$

since $\delta \mathbf{u}_j \equiv 0$. Therefore, $\|g_j\|_\infty \leq \bar{g}_j/2$. Furthermore, for two vertices $(j_1, j_2) and (j_1', j_2') \in [T-1]$, since they both have 4 children:

$$\begin{aligned}
\|\mathbf{u}_{(j_1,j_2)} - \mathbf{u}_{(j_1',j_2')}\|_\infty &\leq \frac{1}{4}\sum_{m=0}^{1}\sum_{n=0}^{1} \|\mathbf{p}((x_{j_1+m}, y_{j_2+n})) - \mathbf{p}((x_{j_1'+m}, y_{j_2'+n}))\| && (7.32)\\
&\leq \frac{1}{4}\sum_{m=0}^{1}\sum_{n=0}^{1} c_S \|(x_{j_1+m}, y_{j_2+n}) - (x_{j_1'+m}, y_{j_2'+n})\| && (7.33)\\
&= \frac{1}{4}\sum_{m=0}^{1}\sum_{n=0}^{1} c_S \max(|j_1 + m - j_1' + m|, |j_2 + m - j_2' + m|) && (7.34)\\
&= c_S \max(|j_1 - j_1'|, |j_2 - j_2'|) = c_S \|\mathbf{l}_{j_1,j_2} - \mathbf{l}_{j_1',j_2'}\| && (7.35)
\end{aligned}$$

Therefore, $\{\mathbf{u}_j\}$ is also a LF-DF with smoothness constant c_S on grid points $\{\mathbf{l}_j\}$, and the grid points satisfies Eqn. 7.24. We thus can apply the proof recursively and Eqn. 7.29 holds for any layer. ∎

Theorem 7.2.7 (HDMM produces LS-DF) *Given any compatible state set $\{(h_j, g_j)\}$ of HDMM, the leaf node locations satisfy:*

$$\|\mathbf{u}_{k_1} - \mathbf{u}_{k_2}\|_\infty \leq 2c_S \|\mathbf{l}_{k_1} - \mathbf{l}_{k_2}\|_\infty \tag{7.36}$$

for any $k_1, k_2 \in [T]$. This shows the leaf locations $\{\mathbf{u}_k\}$ meet the definition of LS-DF with smoothness coefficient $2c_S$.

Proof By Lemma 7.2.4, for $\|\mathbf{u}_{k_1} - \mathbf{u}_{k_2}\|_\infty = l$, there exists a path $P = [k_1, \ldots, j_0, \ldots, k_2]$ that contains $2l+1$ vertices on the hierarchical structure from vertex k_1 to vertex k_2 and $t(j_0) = T - l$. For any parent-child pair (j, k), since $\delta \mathbf{u}_j \equiv 0$ and $(h_j, g_j) \sim h_k$, we have

$$\|\mathbf{u}_j - \mathbf{u}_k\|_\infty = \|g_{jk}\|_\infty \leq \bar{g}_j/2 = c_S \tag{7.37}$$

Using triangle inequality l times, and we obtain $\|\mathbf{u}_{j_0} - \mathbf{u}_{k_1}\|_\infty \leq l c_S$ and $\|\mathbf{u}_{j_0} - \mathbf{u}_{k_2}\|_\infty \leq l c_S$,

and thus:

$$\|\mathbf{u}_{k_1} - \mathbf{u}_{k_2}\|_\infty \leq \|\mathbf{u}_{j_0} - \mathbf{u}_{k_1}\|_\infty + \|\mathbf{u}_{j_0} - \mathbf{u}_{k_2}\|_\infty \leq 2c_S l = 2c_S \|\mathbf{l}_{k_1} - \mathbf{l}_{k_2}\|_\infty \tag{7.38}$$

∎

Note that Theorem 7.2.7 does not hold for tree structured hierarchy. Since any tree structure will have two vertices k_1 and k_2 whose distance is 1 but whose path has length $2T - 1$. As a result, a deformation accepted by any tree-structured HDMM will not satisfy local smoothness condition.

7.3 The Invariant Representation

For any held state $h_j \in \mathcal{H}_j$, the *confidence map* $I_j = I_j(h_j)$ tells the confidence that j-th patch appears with the (rough) pose encoded by h_j. In particular, for a binary map, $I_j(h_j)$ tells whether a part with the given (rough) pose h_j exists on the image.

Interestingly, the confidence map I_j can be used as a *representation* due to its invariant property: I_j only depends on rough state h_j rather than the complete state $\mathbf{p}(S_j)$. Therefore, if the local deformation is changed slightly, $\mathbf{p}(S_j)$ will indeed change but not h_j. This keeps I_j invariant to local changes that play no roles in high-level decisions. Just like our metaphor, I_j is the report sent from Duke's friends that omits detailed affairs but only contains gross incomes.

In contrast, in Top-down Hierarchical Predictions (Chapter 6), representations in all patches are just raw pixels, although in practice, some image blurring and downsampling may be added to improve the Lipschitz constants. Although such a representation is simple, this brings about the concentration of degrees of freedom on the top layers, since one need to handle every details of every pixel of a large raw image, in order to achieve worst-case guarantees for Nearest Neighbor predictors.

Then, how to compute the invariant representation $I_j(\cdot)$ for a patch j? Naively, $I_j(\cdot)$ can be built directly from the raw image I, by enumerating all locations and poses of patch j. Once a candidate pose is found through this exhaustive search, discard all details and only report the rough pose in I_j. This is like how our ex-Baron, Robert, writes his report before he invites his friends. Although the report contains concise information, the procedure of writing it requires knowing all the details.

A more clever way is to *write a report on reports*, which is how our Duke does. He checks the reports from his friends, finds critical parts, and integrates them to write his own version to the King. Similarly, the representation I_j is computed from all its children's representations $I_{ch(j)} \equiv$

$\{I_k\}_{k\in ch(j)}$. Since each representation I_k already discards a lot of details, the computational burden reduces to checking all the rough information contains in $I_{ch(j)}$ and discards those *relative details* from j-th part's point of view.

This yields a much lower computational cost. Throughout the algorithm, the total degrees of freedom d_0 of h_j and g_j will never exceed a manageable limit, and is much smaller than the effective degrees of freedom d of the entire deformation field. The final computational complexity is only exponential to $d_0 \ll d$, breaking the curse of dimensionality.

7.3.1 Computing Representations

Alg. 6 and Alg. 7 are the algorithms used to compute binary invariant representation I_j in a bottom-up fashion. Alg. 6 is the ideal algorithm that leads to concise theoretical analysis. However, an implementation of this algorithm requires enumerating over *infinite* set $\mathcal{H}_j$ and $\mathcal{G}_j$. In practice, we propose its discrete version (Alg. 7) in which only sample points are enumerated. To keep the same theoretical guarantees, an additional *pooling* procedure is introduced.

We list the key updating from child k to parent j in the continuous algorithm as follows:

$$I_{k\to j}(h_j, g_j) = \max_{h_k\sim(h_j,g_j)} I_k(h_k) \quad (Translation) \tag{7.39}$$

$$\tilde{I}_j(h_j, g_j) = \bigvee_{k\in ch(j)} I_{k\to j}(h_j, g_j) \quad (Aggregation) \tag{7.40}$$

$$I_j(h_j) = \max_{g_j\in\mathcal{G}_j} \tilde{I}_j(h_j, g_j) \quad (Reduction) \tag{7.41}$$

where the *aggregation operator* $\bigvee$ is simply the minimal operator $\min$. Intuitively, Eqn. 7.39 collects the information from each child vertex, Eqn. 7.40 aggregates the information and Eqn. 7.41 discards the details to form the representation for the parent vertex j. Fig. 7.4 shows the workflow.

In the discrete case, we simply replace $\sim$ with $\sim_\epsilon$ (which will be introduced in Definition 7.3.5) and evaluate response maps (I_k), on a set of samples with a given density.

Both (upstream) algorithms look very similar to the message-passing procedure on a tree, in which messages go from the leaves to their parents. However, as we shall see, our algorithm has guarantees even in the presence of loopy hierarchy.

7.3.2 Lipschitz Conditions on Representations

It seems that these two algorithms are complicated. However, if the following assumptions hold, then we can have very good theoretical properties for these two algorithms, which serves as the

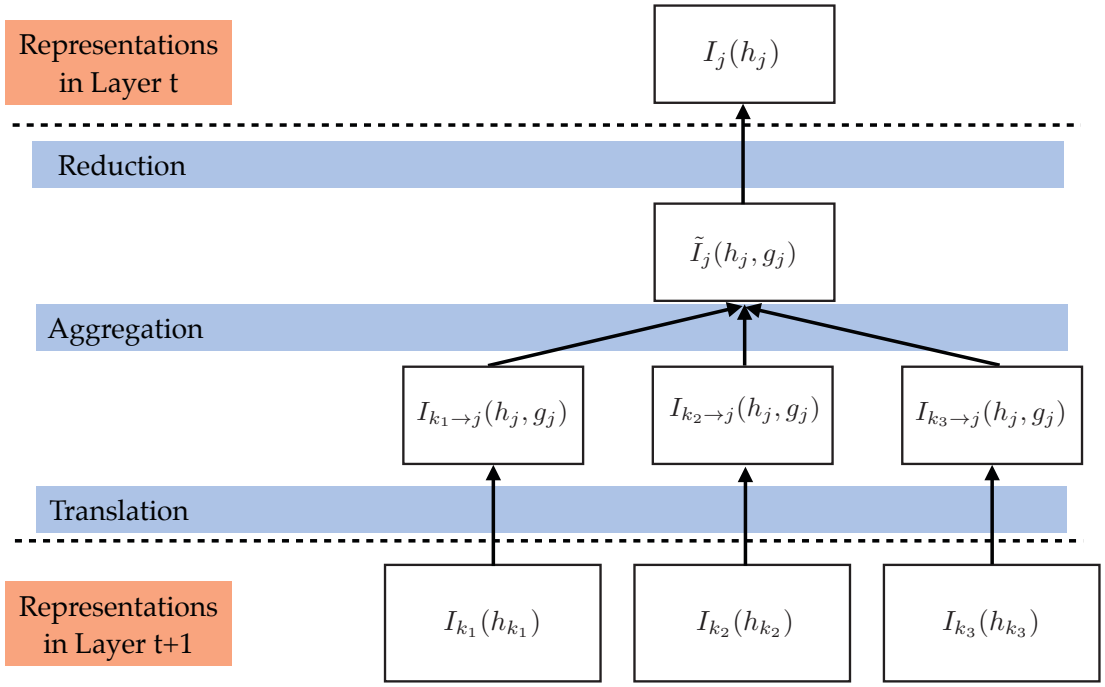


Figure 7.4: Building invariant representations. The parent j first translates the child representation $I_k(h_k)$ to its parental form $I_{k \to j}(h_j, g_j)$ using Eqn. 7.39, aggregates all information from its children to obtain $\tilde{I}_j(h_j, g_j)$ (Eqn. 7.40) and then discards details to obtain its own representation $I_j(h_j)$ (Eqn. 7.41).

Algorithm 6 Computing Representations, Continuous Case

INPUT Raw image I.
OUTPUT Invariant representations $\{I_j\}$.
Compute response map for each landmarks $\{I_k\}_{k=1}^K = \text{InitRep}(I)$.
for $t = T$ downto 1 **do**
  for Vertex $j \in [t]$ **do**
    for State $h_j = (\mathbf{u}_j, z_j) \in \mathcal{H}_j$ **do**
      for Local Displacement $g_j \in \mathcal{G}_j$ **do**
        for Child $k \in ch(j)$ **do**
          $I_{k \to j}(h_j, g_j) = \max_{h_k \sim (h_j, g_j)} I_k(h_k)$.
        end for
      end for
      Let $\tilde{I}_j(h_j, g_j) = \bigvee_{k \in ch(j)} I_{k \to j}(h_j, g_j)$ where $\bigvee = \min$.
      Set $I_j(h_j) = \max_{g_j \in \mathcal{G}_j} \tilde{I}_j(h_j, g_j)$.
    end for
  end for
end for
RETURN $\{I_j\}$.

Algorithm 7 Computing Representations, Discrete Case

INPUT Raw image I.
INPUT Samples $(\mathbf{u}_j^i) \in \mathcal{H}_j$ and $g_j^i \in \mathcal{G}_j^s$ with density parameters ϵ_k^h and ϵ_k^g defined in Eqn. 7.81.
OUTPUT Invariant representations $\{I_j\}$.
Compute response map for each landmarks $\{I_k\}_{k=1}^K = \text{InitRep}(I)$.
for $t = T$ downto 1 **do**
  for Vertex $j \in [t]$ **do**
    for $i_1 = 1$ to $\#\{\mathbf{u}_j^i\}$ **do**
      for $z_j \in \mathcal{Z}_j$ **do**
        for $i_2 = 1$ to $\#\{g_j^i\}$ **do**
          Set $h_j^i = (\mathbf{u}_j^{i_1}, z_j)$.
          for Child $k \in ch(j)$ **do**
            Compute (See Eqn. 7.7): $\tilde{\mathbf{u}}_k = \mathbf{u}_j^{i_1} + \delta\mathbf{u}_{jk}(z_j) + g_{jk}^{i_2}$.
            Pooling. Set

$$I_{k\to j}(\mathbf{u}_j^{i_1}, z_j, g_j^{i_2}) = \max_{\|\mathbf{u}_k^{i'} - \tilde{\mathbf{u}}_k\| \le \epsilon_k^h} \max_{z_k \sim z_j} I_k(\mathbf{u}_k^{i'}, z_k) \tag{7.42}$$

$$= \max_{\text{sampled } h_k^{i'} \sim_\epsilon (h_j^i, g_j^{i_2})} I_k(h_k^{i'}) \tag{7.43}$$

          end for
        end for
        Let $\tilde{I}_j(\mathbf{u}_j^{i_1}, z_j, g_j^{i_2}) = \bigvee_{k \in ch(j)} I_{k\to j}(\mathbf{u}_j^{i_1}, z_j, g_j^{i_2})$ where $\bigvee = \min$.
        Set $I_j(\mathbf{u}_j^{i_1}, z_j) = \max_{i_2} \tilde{I}_j(\mathbf{u}^{i_1}, z_j, g_j^{i_2})$.
      end for
    end for
  end for
end for
RETURN $\{I_j\}$.

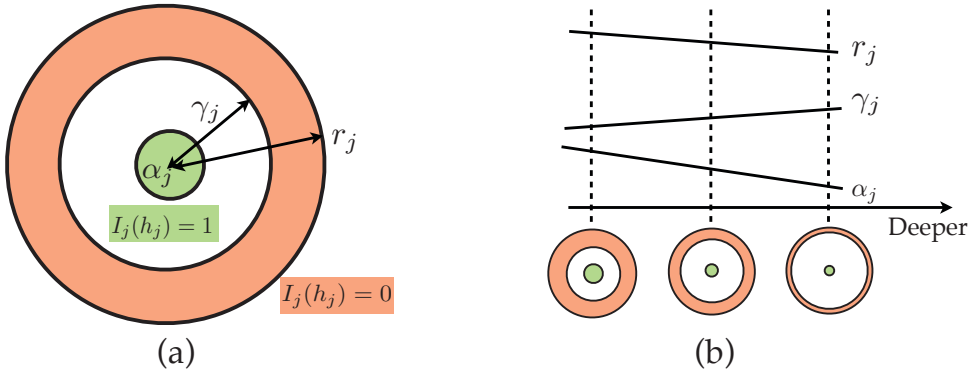


Figure 7.5: The constants $(\alpha_j, \gamma_j, r_j)$ in the Lipschitz Condition (Assumption 7.3.1). **(a)** Within radius α_j of the true state h_j, $I_j(h_j) = 1$; for h_j that is within the radius of r_j but out of the radius γ_j, $I_j(h_j) = 0$. For the remaining regions, $I_j(h_j)$ is not determined. **(b)** How the constants change with respect to the depth. From bottom to top, α_j and r_j decrease while γ_j increases (See Eqn. 7.68, Eqn. 7.69, Eqn. 7.70). As long as the condition (Eqn. 7.110) holds, the global optimality of Alg. 8 is guaranteed.

building block of our bottom-up deformation estimation algorithm:

Assumption 7.3.1 (Lipschitz Conditions of Landmarks) *Given an image $I_{\mathbf{p}}$ that is generated from the parameter $\mathbf{p}$, it is assumed there is a procedure called* InitRep *that compute a 2D response map I_k for each landmark k so that there exists a 3-tuples $\alpha_j < \gamma_j < r_j$:*

- ***Smooth:*** *For any $\mathbf{v}$ with $\|\mathbf{p}(k) - \mathbf{v}\|_\infty \leq \alpha_k$, $I_k(\mathbf{v}) = 1$. As a special case, $I_k(\mathbf{p}(k)) = 1$.*
- ***Local Distinctive:*** *For any $\mathbf{v}$ with $r_k \geq \|\mathbf{p}(k) - \mathbf{v}\|_\infty \geq \gamma_k$, $I_k(\mathbf{v}) = 0$.*

Note that in this assumption, if $r_k = +\infty$, then the landmark k becomes *globally* distinctive. This means that it can be independently identified without knowing the location of other landmarks. Feature matching approaches, such as SIFT, implicitly assumes $r_k = +\infty$ for key points and thus has stronger assumptions compared to Assumption 7.3.1.

Given this assumption, we now proceed to show that the representation computed from Alg. 6 and Alg. 7 are well-behaved. To achieve that, we consider an image $I_{\mathbf{p}}$ that is generated from the parameter $\mathbf{p}$. Denote $h_j^* = (\mathbf{u}_j^*, z_j^*)$ as the derived parameter set computed from Eqn. 7.7 and Eqn. 7.8, and $h_j \in \mathcal{H}_j$ as an arbitrary held state. We define the distance between two held state h_j and h_j' as:

$$\text{dist}(h_j, h_j') = \|\mathbf{u}_j - \mathbf{u}_j'\|_\infty + \|\delta\mathbf{u}_j(z_j) - \delta\mathbf{u}_j(z_j')\|$$

where $\|\delta\mathbf{u}_j(z_j) - \delta\mathbf{u}_j(z_j')\|$ is defined in Eqn. 7.10.

It is easy to verify the distance $\text{dist}(h_j, h_j^*)$ satisfies the requirement of metric:

- $\text{dist}(h_j, h_j') \geq 0$ and $\text{dist}(h_j, h_j') = \text{dist}(h_j, h_j')$. That is, the distance is nonnegative and symmetric.

- $\mathrm{dist}(h_j, h'_j) = 0$ if and only if $h_j = h'_j$, i.e., $\mathbf{u}_j = \mathbf{u}'_j$ and $z_j = z'_j$. This can be proven from nondegenerative condition of types (Eqn. 7.9).
- The triangle inequality for $\mathrm{dist}(h_j, h'_j)$ follows naturally from the metric property of $\|\delta\mathbf{u}_j(z_j) - \delta\mathbf{u}_j(z'_j)\|$.

In addition, the distance metric satisfies the following lemma, which serves as the building block of our analysis:

Lemma 7.3.2 (Parent-Child Bound) *Suppose we have state h_j, (h'_j, g'_j) and h'_k with the condition $\mathrm{dist}(h_j, h'_j) \leq \lambda$ and $(h'_j, g'_j) \sim h'_k$. Then for any child $k \in ch(j)$, any child type $z_k \sim z_j$ and local displacement g_j, the constructed held state $h_k = (\mathbf{u}_k, z_k) \sim (h_j, g_j)$ satisfies:*

$$\mathrm{dist}(h_k, h'_k) \leq \phi^+_{jk}(\lambda) + \|g_j - g'_j\|_\infty \tag{7.44}$$

Proof The premise $\mathrm{dist}(h_j, h'_j) \leq \lambda$ means

$$\|\mathbf{u}'_j - \mathbf{u}_j\|_\infty + \|\delta\mathbf{u}_j(z'_j) - \delta\mathbf{u}_{jk}(z_j)\|_\infty \leq \lambda \tag{7.45}$$

Therefore, we have for any $k \in ch(j)$ and *any* $g_j \in \mathcal{G}_j$:

$$\begin{aligned}
\|\mathbf{u}_k - \mathbf{u}'_k\|_\infty &= \|\mathbf{u}_j + \delta\mathbf{u}_{jk}(z_j) + g_{jk} - (\mathbf{u}'_j + \delta\mathbf{u}_{jk}(z'_j) + g'_{jk})\|_\infty && (7.46)\\
&\leq \|\mathbf{u}_j - \mathbf{u}'_j\|_\infty + \|\delta\mathbf{u}_{jk}(z_j) - \delta\mathbf{u}_{jk}(z'_j)\|_\infty + \|g_{jk} - g'_{jk}\|_\infty && (7.47)\\
&\leq \mathrm{dist}(h_j, h'_j) + \|g_{jk} - g'_{jk}\| && (7.48)\\
&\leq \lambda + \|g_j - g'_j\|_\infty && (7.49)
\end{aligned}$$

On the other hand, from Eqn. 7.45, we have:

$$\|\delta\mathbf{u}_j(z'_j) - \delta\mathbf{u}_j(z_j)\| \leq \lambda \tag{7.50}$$

Therefore, using the definitions in Eqn. 7.21, we have:

$$\|\delta\mathbf{u}_k(z_k) - \delta\mathbf{u}_k(z'_k)\| \leq \delta\bar{\mathbf{u}}^+_{jk}(\lambda) \tag{7.51}$$

Combining them together, if k is not a leaf vertex, we have for *any* $z_k \in \psi^+_{jk}(z_j)$:

$$\begin{aligned}
\mathrm{dist}(h_k, h'_k) &= \|\mathbf{u}_k - \mathbf{u}'_k\| + \|\delta\mathbf{u}_k(z_k) - \delta\mathbf{u}_k(z'_k)\| && (7.52)\\
&\leq \lambda + \|g_j - g'_j\|_\infty + \delta\bar{\mathbf{u}}^+_{jk}(\lambda) && (7.53)\\
&= \phi^+_{jk}(\lambda) + \|g_j - g'_j\|_\infty && (7.54)
\end{aligned}$$

If k is a leaf vertex, then $\delta\bar{\mathbf{u}}^+_{jk}(\lambda)$ is simply zero and ϕ^+_{jk} is the identity function. The inequality still holds. ∎

Lemma 7.3.3 (Parent-Child Bound #2) *Since we have two pairs of states $h_k \sim (h_j, g_j)$, $h'_k = (\mathbf{u}'_k, z'_k)$ and $h'_j = (\mathbf{u}'_j, z'_j)$ with $z'_j \sim z'_k$. In addition, we have the condition $\mathrm{dist}(h_k, h'_k) \leq \lambda$, then we have:*

$$\mathrm{dist}(h_j, h'_j) \leq \phi^-_{jk}(\lambda) + \frac{\bar{g}_j}{2} + \|\mathbf{u}'_j + \delta\mathbf{u}_{jk}(z'_j) - \mathbf{u}'_k\|_\infty \tag{7.55}$$

In particular, if $h'_k \sim (h'_j, g'_j)$, then

$$\mathrm{dist}(h_j, h'_j) \leq \phi^-_{jk}(\lambda) + \bar{g}_j \tag{7.56}$$

Proof Note that the condition $\mathrm{dist}(h_k, h'_k) \leq \lambda$ means:

$$\|\mathbf{u}_k - \mathbf{u}'_k\|_\infty \leq \lambda \quad \wedge \quad \|\delta\mathbf{u}_k(z_k) - \delta\mathbf{u}_k(z'_k)\| \leq \lambda \tag{7.57}$$

From the definition of Eqn. 7.22, we have:

$$\|\delta\mathbf{u}_j(z_j) - \delta\mathbf{u}_j(z'_j)\| \leq \delta\bar{\mathbf{u}}^-(\lambda) \tag{7.58}$$

On the other hand, denote $\tilde{g}_{jk} = \mathbf{u}'_k - \mathbf{u}'_j - \delta\mathbf{u}_{jk}(z'_j)$, then $\mathbf{u}'_j = \mathbf{u}'_k - \delta\mathbf{u}_{jk}(z'_j) - \tilde{g}_{jk}$, and we have:

$$\begin{aligned}
\|\mathbf{u}_j - \mathbf{u}'_j\|_\infty &= \|\mathbf{u}_k - \delta\mathbf{u}_{jk}(z_j) - g_{jk} - (\mathbf{u}'_k - \delta\mathbf{u}_{jk}(z'_j) - \tilde{g}_{jk})\|_\infty && (7.59)\\
&\leq \|\mathbf{u}_k - \mathbf{u}'_k\|_\infty + \|\delta\mathbf{u}_{jk}(z'_j) - \delta\mathbf{u}_{jk}(z_j)\|_\infty + \|g_{jk}\|_\infty + \|\tilde{g}_{jk}\|_\infty && (7.60)\\
&\leq \|\mathbf{u}_k - \mathbf{u}'_k\|_\infty + \|\delta\mathbf{u}_j(z'_j) - \delta\mathbf{u}_j(z_j)\| + \|g_{jk}\|_\infty + \|\tilde{g}_{jk}\|_\infty && (7.61)\\
&\leq \lambda + \delta\bar{\mathbf{u}}^-(\lambda) + \frac{\bar{g}_j}{2} + \|\tilde{g}_{jk}\|_\infty && (7.62)
\end{aligned}$$

Combining the two together, and we have:

$$\begin{aligned}
\mathrm{dist}(h_j, h'_j) &= \|\mathbf{u}_j - \mathbf{u}'_j\|_\infty + \|\delta\mathbf{u}_j(z_j) - \delta\mathbf{u}_j(z'_j)\| && (7.63)\\
&\leq \lambda + 2\delta\bar{\mathbf{u}}^-(\lambda) + \frac{\bar{g}_j}{2} + \|\tilde{g}_{jk}\|_\infty && (7.64)\\
&= \phi^-_{jk}(\lambda) + \frac{\bar{g}_j}{2} + \|\mathbf{u}'_j + \delta\mathbf{u}_{jk}(z'_j) - \mathbf{u}'_k\|_\infty && (7.65)
\end{aligned}$$

∎

Lipschitz Conditions, Continuous Case

With the help of Lemma 7.3.2, we thus proceed to prove the main theorem for invariant representations. We start from the continuous case (Alg. 6) which is not implementable but easier to understand and analyze.

Suppose we have an image $I_{\mathbf{p}}$ that is generated from the parameter $\mathbf{p}$. Denote $h_j^* = (\mathbf{u}_j^*, z_j^*)$ as the derived parameter set computed from Eqn. 7.7 and Eqn. 7.8, and $h_j \in \mathcal{H}_j$ as an arbitrary held state.

In the continuous case, Thm. 7.3.4 shows that each layer of the invariant representations computed from raw image $I = I_{\mathbf{p}}$ are all well-behaved, if representations are constructed using Alg. 6.

Fig. 7.5 shows graphically how $I_j(\mathbf{u}_j, z_j)$ looks like. In the local neighborhood ($\mathrm{dist}(h_j, h_j^*) \leq \alpha_j$) of the true location h_j^*, the responses remain positive. If we step away from the true location, in the region of ring ($\gamma_j \leq \mathrm{dist}(h_j, h_j^*) \leq r_j$), the responses must be negative. In the gray region $\alpha_j < \mathrm{dist}(h_j, h_j^*) < \gamma_j$, one cannot guarantee the responses. Here the constants $(\alpha_j, \gamma_j, r_j)$ are built from bottom to up.

Theorem 7.3.4 (Invariant Representations, Continuous Case) *Then if Assumption 7.3.1 holds for all landmarks, for every part j, using Alg. 6, we have:*

- ***Smooth:*** *If a state h_j is sufficiently close to the "true" h_j^*, then h_j is also plausible:*

$$\mathrm{dist}(h_j, h_j^*) \leq \alpha_j \implies I_j(h_j) = 1 \tag{7.66}$$

- ***Local Distinctive:*** *If h_j is far away from the "true" h_j^* but is not too far, then h_j is not plausible:*

$$r_j \geq \mathrm{dist}(h_j, h_j^*) \geq \gamma_j \implies I_j(h_j) = 0 \tag{7.67}$$

if the 3-tuples $(\alpha_j, \gamma_j, r_j)$ are built for j-th part in a bottom-up fashion, using the following relationship ($ah(j)$ are the anchor points of j-th part):

$$\alpha_j = \min_{k \in ch(j)} (\phi_{jk}^+)^{-1}(\alpha_k) \tag{7.68}$$

$$\gamma_j = \min_{k \in ah(j)} \phi_{jk}^-(\gamma_k) + \bar{g}_j \tag{7.69}$$

$$r_j = \min_{k \in ah(j)} (\phi_{jk}^+)^{-1}(r_k - \bar{g}_j) \tag{7.70}$$

Proof We prove the theorem by induction. The base case follows naturally from Assumption 7.3.1 with $|\mathcal{Z}_k| = 1$ for all leaf vertex k. Now let us prove that this theorem works for any vertex j given all its children follows the theorem.

Smooth: Using Lemma 7.3.2, from the premise $\mathrm{dist}(h_j, h_j^*) \leq \alpha_j$, for any $k \in ch(j)$ and *any* $z_k \sim z_j$, we can construct $h_k = (\mathbf{u}_k, z_k)$ with $g_j = g_j^*$ so that:

$$\mathrm{dist}(h_k, h_k^*) \;\; \leq \;\; \phi_{jk}^+(\alpha_j) \leq \alpha_k \tag{7.71}$$

The last inequality is due to the fact that $\phi_{jk}^+(\cdot)$ is a monotonously increasing function. By induction, $I_k(h_k) = I_k(\mathbf{u}_k, z_k) = 1$ for any $k \in ch(j)$ and any $z_k \sim z_j$. Therefore, for the particular $(\mathbf{u}_k, z_k) = h_k \sim (h_j, g_j)$ for any k:

$$I_{k\to j}(\mathbf{u}_j, z_j, g_j) = I_{k\to j}(h_j, g_j) = \max_{h_k' \sim (h_j, g_j)} I_k(h_k') \geq I_k(h_k) = 1 \tag{7.72}$$

which means

$$\tilde{I}_j(\mathbf{u}_j, z_j, g_j) = \min_{k \in ch(j)} I_{k\to j}(\mathbf{u}_j, z_j, g_j) = 1 \tag{7.73}$$

Since for this particular g_j, $\tilde{I}_j(\mathbf{u}_j, z_j, g_j) = 1$. we have:

$$1 \geq I_j(\mathbf{u}_j, z_j) = \max_{g_j \in \mathcal{G}_j} \tilde{I}_j(\mathbf{u}_j, z_j, g_j) \geq 1 \tag{7.74}$$

Therefore, $I_j(h_j) = I_j(\mathbf{u}_j, z_j) = 1$.

Local Distinctive: We prove this part by contradiction, i.e., if $\mathrm{dist}(h_j, h_j^*) \leq r_j$ and $I_j(h_j) = 1$, then $\mathrm{dist}(h_j, h_j^*) \leq \gamma_j$.

Since $I_j(h_j) = 1$, there exists g_j so that

$$\tilde{I}_j(\mathbf{u}_j, z_j, g_j) = \min_{k \in ch(j)} I_{k\to j}(\mathbf{u}_j, z_j, g_j) = 1 \tag{7.75}$$

Therefore, for any k, we have

$$I_{k\to j}(\mathbf{u}_j, z_j, g_j) = \max_{h_k \sim (h_j, g_j)} I_k(h_k) = 1 \tag{7.76}$$

which means there exists $(\mathbf{u}_k, z_k) = h_k \sim (h_j, g_j)$ so that

$$I_k(h_k) = I_k(\mathbf{u}_k, z_k) = 1 \tag{7.77}$$

Using Lemma 7.3.2, for the anchor points $k \in ah(j)$, for $z_k \sim z_j$, for g_{jk} and their associated h_k, we have:

$$\mathrm{dist}(h_k, h_k^*) \leq \phi_{jk}^+(r_j) + \|g_j - g_j^*\| \leq \phi_{jk}^+(r_j) + \bar{g}_j \leq r_k \tag{7.78}$$

where the final inequality is due to the condition (Eqn. 7.70). By induction, for all $k \in ah(j)$, we have $\mathrm{dist}(h_k, h_k^*) \leq \gamma_k$. Note that by definition $h_k \sim (h_j, g_j)$ and $h_k^* \sim (h_j^*, g_j^*)$, then using Lemma 7.3.3 and we have:

$$\mathrm{dist}(h_j, h_j^*) \leq \phi_{jk}^-(\gamma_k) + \bar{g}_k \tag{7.79}$$

This works for any $k \in ah(j)$, therefore, we have:

$$\mathrm{dist}(h_j, h_j^*) \leq \min_{k \in ah(j)} \phi_{jk}^-(\gamma_k) + \bar{g}_k = \gamma_j \tag{7.80}$$

∎

7.3.3 Lipschitz Conditions: Discrete Case

Obviously, one cannot enumerate over an infinite set $\mathcal{H}_j$ or $\mathcal{G}_j$, which leads to infinite sample complexity. Then the problem becomes, can the similar guarantee be obtained if we just sample from the infinite sets $\mathcal{H}_j$ and $\mathcal{G}_j$? Fortunately, as indicated by Thm. 7.3.7, if the sample density is high enough, the answer is yes with a modified algorithm (Alg. 7).

For every part j, we sample the space $\mathcal{H}_j$ and $\mathcal{G}_j$ so that for every $\mathbf{u}_j$ and every $g_j \in \mathcal{G}_j$, there exists at least one sample $\mathbf{u}_j^{i_1}$ and $g_j^{i_2}$ that are close to them:

$$\|\mathbf{u}_j^{i_1} - \mathbf{u}_j\|_\infty \leq \epsilon_j^h, \quad \|g_j^{i_2} - g_j\|_\infty \leq \epsilon_j^g \tag{7.81}$$

Note $\mathcal{Z}_j$ need not be sampled since it is a discrete space. We can also define the compatibility symbol $\sim_\epsilon$ in the discrete case:

Definition 7.3.5 $(h_j, g_j) \sim_\epsilon h_k$ *if and only if* $z_j \sim z_k$ *and* $\|\mathbf{u}_j + \delta\mathbf{u}_{jk}(z_j) + g_{jk} - \mathbf{u}_k\| \leq \epsilon_k^h$

Given these sampling strategy, we can prove the following lemma, which is an extension for Lemma 7.3.2:

Lemma 7.3.6 (Parent-Child Bound, With Samples) *Suppose we have a sample point* h_j^i*, a state* (h_j', g_j') *and* h_k' *with* $\mathrm{dist}(h_j^i, h_j') \leq \lambda$ *and* $(h_j', g_j') \sim h_k'$*. For any local displacement* g_j*, for any child* $k \in ch(j)$*, any sampled state* $h_k^{i'} = (\mathbf{u}_k^{i'}, z_k) \sim_\epsilon (h_j^i, g_j)$ *satisfies*

$$\mathrm{dist}(h_k^{i'}, h_k') \leq \phi_{jk}^+(\lambda) + \epsilon_k^h + \|g_j - g_j'\|_\infty \tag{7.82}$$

Proof Using Lemma 7.3.2, since $\mathrm{dist}(h_j^i, h_j') \leq \lambda$, for any $k \in ch(j)$ and *any* $z_k \sim z_j$ and any g_j, we can construct a state $h_k = (\tilde{\mathbf{u}}_k, z_k) \sim (h_j^i, g_j)$ so that:

$$\mathrm{dist}(h_k, h_k^*) \quad \leq \quad \phi_{jk}^+(\lambda) + \|g_j - g_j^*\|_\infty \tag{7.83}$$

with $\tilde{\mathbf{u}}_k = \mathbf{u}_j^i + \delta\mathbf{u}_{jk}(z_j) + g_{jk}$. Note that $\tilde{\mathbf{u}}_k$ is not necessarily within the samples. However, since the sample state $h_k^{i'} = (\mathbf{u}_k^{i'}, z_k)$ is ϵ-compatible with (h_j^i, g_j), we have

$$\|\tilde{\mathbf{u}}_k - \mathbf{u}_k^{i'}\|_\infty \leq \epsilon_k^h \tag{7.84}$$

Therefore, for $h_k = (\tilde{\mathbf{u}}_k, z_k)$ and $h_k^{i'} = (\mathbf{u}_k^{i'}, z_k)$, we apply triangle inequality and obtain:

$$\text{dist}(h_k^{i'}, h_k^*) \leq \text{dist}(h_k^{i'}, h_k) + \text{dist}(h_k, h_k^*) \leq \phi_{jk}^+(\lambda) + \epsilon_k^h + \|g_j - g_j^*\|_\infty \tag{7.85}$$

We thus have the following theorem:

Theorem 7.3.7 (Invariant Representations, Discrete Case) *If Assumption 7.3.1 holds for all landmarks and the sample density satisfies Eqn. 7.81, then for every part j, using Alg. 7, we have:*

- ***Smooth:*** *If a* sampled state h_j^i *is sufficiently close to the "true" h_j^*, then h_j^i is also plausible:*

$$\text{dist}(h_j^i, h_j^*) \leq \alpha_j \implies I_j(h_j^i) = 1 \tag{7.86}$$

- ***Local Distinctive:*** *If a* sampled state h_j^i *is far away from the "true" h_j^* but is not too far, then h_j^i is not plausible:*

$$r_j \geq \text{dist}(h_j^i, h_j^*) \geq \gamma_j \implies I_j(h_j^i) = 0 \tag{7.87}$$

if the 3-tuples $(\alpha_j, \gamma_j, r_j)$ are built for j-th part in a bottom-up fashion, using the following relationship ($ah(j)$ are the anchor points of j-th part):

$$\alpha_j = \min_{k \in ch(j)} (\phi_{jk}^+)^{-1}(\alpha_k - \epsilon_j^g - \epsilon_k^h) \tag{7.88}$$

$$\gamma_j = \min_{k \in ah(j)} \phi_{jk}^-(\gamma_k) + \bar{g}_j + \epsilon_k^h \tag{7.89}$$

$$r_j = \min_{k \in ah(j)} (\phi_{jk}^+)^{-1}(r_k - \bar{g}_j - \epsilon_k^h) \tag{7.90}$$

Proof We prove the theorem by induction. The base case follows naturally from Assumption 7.3.1 with $|\mathcal{Z}_k| = 1$ for all leaf vertex k. In addition, for any sample distribution, the smoothness and local distinctiveness conditions will hold.

Now let us prove that this theorem works for any vertex j given all its children follows the theorem.

Smooth: From Eqn. 7.81, there exists $g_j^{i_2}$ so that $\|g_j^{i_2} - g_j^*\|_\infty \leq \epsilon_j^g$. For any sampled state h_j^i and the particular local displacement $g_j^{i_2}$, for every child k, by the sample density condition

(Eqn. 7.81), there exists $\mathbf{u}_k^{i'}$ so that

$$\|\mathbf{u}_k^{i'} - \tilde{\mathbf{u}}_k\|_\infty \leq \epsilon_k^h \tag{7.91}$$

where $\tilde{\mathbf{u}}_k = \mathbf{u}_j^i + \delta\mathbf{u}_{jk}(z_j) + g_{jk}^{i_2}$. Therefore, we can construct a sampled state $(\mathbf{u}_k^{i'}, z_k) = h_k^{i'} \sim_\epsilon (h_j^i, g_j^{i_2})$. Using Lemma 7.3.6, we have:

$$\text{dist}(h_k^{i'}, h_k^*) \leq \phi_{jk}^+(\alpha_j) + \epsilon_k^h + \|g_j^{i_2} - g_j'\|_\infty \leq \phi_{jk}^+(\alpha_j) + \epsilon_k^h + \epsilon_j^g \leq \alpha_k \tag{7.92}$$

The last inequality is due to the fact that $\phi_{jk}^+(\cdot)$ is a monotonously increasing function. By induction,

$$I_k(h_k^{i'}) = I_k(\mathbf{u}_k^{i'}, z_k) = 1 \tag{7.93}$$

holds for any k. Since $h_k^{i'} \sim_\epsilon (h_j^i, g_j^{i_2})$, from the pooling operation Eqn. 7.43, we have:

$$I_{k\to j}(h_j^i, g_j^{i_2}) = \max_{h_k^{i'} \sim_\epsilon (h_j^i, g_j^{i_2})} I_k(h_k^{i'}) = 1 \tag{7.94}$$

holds for any k. and thus

$$\tilde{I}_j(h_j^i, g_j^{i_2}) = \min_{k \in ch(j)} I_{k\to j}(h_j^i, g_j^{i_2}) = 1 \tag{7.95}$$

Therefore, we have

$$I_j(h_j^i) = \max_{i_2} \tilde{I}_j(h_j^i, g_j^{i_2}) = 1 \tag{7.96}$$

Local Distinctive: We prove this part by contradiction, i.e., if $\text{dist}(h_j^i, h_j^*) \leq r_j$ and $I_j(h_j^i) = 1$, then $\text{dist}(h_j^i, h_j^*) \leq \gamma_j$.

Firstly, using Lemma 7.3.2, for any anchor point $k \in ah(j)$, for any $z_k \sim z_j$, any g_{jk}, and their associated $\tilde{\mathbf{u}}_k$

$$\tilde{\mathbf{u}}_k = \mathbf{u}_j^i + \delta\mathbf{u}_{jk}(z_j) + g_{jk} \tag{7.97}$$

and $\tilde{h}_k = (\tilde{\mathbf{u}}_k, z_k)$, we have:

$$\text{dist}(\tilde{h}_k, h_k^*) \leq \phi_{jk}^+(r_j) + \bar{g}_j \leq r_k - \epsilon_k^h \tag{7.98}$$

Since $I_j(h_j^i) = 1$, there exists $g_j^{i_2}$ so that

$$\tilde{I}_j(\mathbf{u}_j^i, z_j, g_j) = \min_{k \in ch(j)} I_{k\to j}(\mathbf{u}_j^i, z_j, g_j^{i_2}) = 1 \tag{7.99}$$

which also holds for the anchor point $k \in ch(j)$. Therefore,

$$I_{k\to j}(\mathbf{u}_j^i, z_j, g_j^{i_2}) = \max_{h_k^{i'} \sim_\epsilon (h_j^i, g_j^{i_2})} I_k(h_k^{i'}) = 1 \tag{7.100}$$

which means there exists $(\mathbf{u}_k^{i'}, z_k) = h_k^{i'} \sim_\epsilon (h_j^i, g_j^{i_2})$ so that

$$I_k(h_k^{i'}) = I_k(\mathbf{u}_k^{i'}, z_k) = 1 \tag{7.101}$$

From the definition of $\sim_\epsilon$, we thus have:

$$\|\mathbf{u}_k^{i'} - \tilde{\mathbf{u}}_k\|_\infty \leq \epsilon_k^h \tag{7.102}$$

Now we have:

$$\mathrm{dist}(h_k^{i'}, h_k^*) \leq \mathrm{dist}(h_k^{i'}, \tilde{h}_k) + \mathrm{dist}(\tilde{h}_k, h_k^*) \leq \|\mathbf{u}_k^{i'} - \tilde{\mathbf{u}}_k\|_\infty + \mathrm{dist}(\tilde{h}_k, h_k^*) \leq r_k \tag{7.103}$$

By induction, we have $\mathrm{dist}(h_k^{i'}, h_k^*) \leq \gamma_k$.

On the other hand, combining Eqn. 7.102 and Eqn. 7.97, we have:

$$\begin{aligned} \|\mathbf{u}_j^i + \delta\mathbf{u}_{jk}(z_j) - \mathbf{u}_k^{i'}\| &\leq \|\mathbf{u}_j^i + \delta\mathbf{u}_{jk}(z_j) + g_{jk} - \mathbf{u}_k^{i'}\|_\infty + \|g_{jk}\|_\infty & (7.104)\\ &= \|\tilde{\mathbf{u}}_k - \mathbf{u}_k^{i'}\|_\infty + \|g_{jk}\|_\infty & (7.105)\\ &\leq \frac{\bar{g}_j}{2} + \epsilon_k^h & (7.106) \end{aligned}$$

Note that $h_k^{i'} \sim_\epsilon (h_j^i, g_j^{i_2})$ and $h_k^* \sim (h_j^*, g_j^*)$, we then apply Lemma 7.3.3 and obtain:

$$\mathrm{dist}(h_j, h_j^*) \leq \phi_{jk}^-(\gamma_k) + \bar{g}_k + \epsilon_k^h \tag{7.107}$$

This works for any $k \in ah(j)$, therefore, we have:

$$\mathrm{dist}(h_j, h_j^*) \leq \min_{k\in ah(j)} \phi_{jk}^-(\gamma_k) + \bar{g}_k + \epsilon_k^h = \gamma_j \tag{7.108}$$

∎

7.4 The Algorithm

Now we have studied the properties of invariant representations. From the representation, we naturally arrive at Alg. 8 for estimating the location of landmarks.

Algorithm 8 Bottom-Up Deformation Estimation

INPUT: Input Raw Image I.
INPUT: Initial root guess $\tilde{h}_{\text{root}}$ so that $\text{dist}(\tilde{h}_{\text{root}}, h^*_{\text{root}}) = r_{\text{root}}/2$.
OUTPUT: $\{\hat{h}_j\}$ for every part j.
From raw image I, Compute invariant representations $\{I_j\}$ using Alg. 7.
Search the circle with radius $r_{\text{root}}/2$ centered at $\tilde{h}_{\text{root}}$ and find $\hat{h}_{\text{root}}$ so that $I_{\text{root}}(\hat{h}_{\text{root}}) = 1$.(Thm. 7.4.2)
for $t = 1$ to T **do**
  for Vertex $j \in [t]$ **do**
    Consistency Check: if there exists l_1 and l_2 so that

$$\text{dist}(\hat{h}_{l_1 \to j}, \hat{h}_{l_1 \to j}) > 2\gamma_j \tag{7.109}$$

    Then **RETURN** FAIL.
    Randomly pick parent l and set $\hat{h}_j = \hat{h}_{l \to j}$. From Thm. 7.4.1, $I_j(\hat{h}_j) = 1$.
    Obtain optimal $\hat{g}_j$ associated with $\hat{h}_j$ from the bottom-up procedure.
    for Child $k \in pa(j)$ **do**
      Find $h^{i'}_k \sim_\epsilon (\hat{h}_j, \hat{g}_j)$ so that $I_k(h^{i'}_k) = 1$. Thm 7.4.1 guarantees its existence.
      Set $\hat{h}_{j \to k} = \hat{h}^i_k$.
    end for
  end for
end for
RETURN $\{\hat{h}_j, \hat{g}_j\}$.

7.4.1 Global Optimal Guarantees Under Generative Models

Now we show the global optimal guarantees of Alg. 8 if the input image $I = I_{\mathbf{p}}$ is generated by displacements $\mathbf{p}$ of k landmarks.

Lemma 7.4.1 (Top-down Pass of Correct Parameters) *If the following condition holds for every parent-child pair $k \in ch(j)$:*

$$\phi^+_{jk}(\gamma_j) + \bar{g}_j + \epsilon^h_k \leq r_k \tag{7.110}$$

Then for j-th patch, if there is a sampled point $h_j^i = (\mathbf{u}_j^i, z_j)$ satisfying:

$$\mathrm{dist}(h_j^i, h_j^*) \le r_j \quad \wedge \quad I_j(h_j^i) = 1 \tag{7.111}$$

then there exists a sampled point $g_j^{i_2} \in \mathcal{G}_j$ so that for every child k, we can find a sampled point $h_k^{i'} = (\mathbf{u}_k^{i'}, z_k) \sim_\epsilon (h_j^i, g_j^{i_2})$ so that:

$$\mathrm{dist}(h_k^{i'}, h_k^*) \le r_k \quad \wedge \quad I_k(h_k^{i'}) = 1 \tag{7.112}$$

Proof Since $I_j(h_j^i) = 1$, according to the construction of invariant representation (Alg. 7), there exists $g_j^{i_2}$ so that for every k, we can find at least one $(\mathbf{u}_k^{i'}, z_k) = h_k^{i'} \sim_\epsilon (h_j^i, g_j^{i_2})$ so that $I_k(h_k^{i'}) = 1$. Now we need to prove such $h_k^{i'}$ is close to h_k^*.

From Thm. 7.3.7, if $\mathrm{dist}(h_j^i, h_j^*) \le r_j$ and $I_j(h_j^i) = 1$, then $\mathrm{dist}(h_j^i, h_j^*) \le \gamma_j$. By Lemma 7.3.6, for the particular $g_j^{i_2}$ and particular z_k, $h_k^{i'} \sim_\epsilon (h_j^i, g_j^{i_2})$ satisfies:

$$\mathrm{dist}(h_k^{i'}, h_k^*) \le \phi_{jk}^+(\gamma_j) + \|g_j^{i_2} - g_j^*\|_\infty + \epsilon_k^h \le \phi_{jk}^+(\gamma_j) + \bar{g}_j + \epsilon_k^h \le r_k \tag{7.113}$$

∎

Then we present the main theorem that shows Alg. 8 indeed gives

Theorem 7.4.2 (Correctness of Bottom-Up Deformation Estimation (Alg. 8)) *Given $I = I_{\mathbf{p}}$ as input, if*

- *The initial guess $\tilde{h}_{root}$ is close to h_{root}^*: $\mathrm{dist}(\tilde{h}_{root}, h_{root}^*) \le r_{root}/2$.*
- *The sample density on the root vertex satisfies the following:*

$$\epsilon_{root}^h \le \alpha_{root} \tag{7.114}$$

- *The following inequality holds for any parent-child pair (j, k),*

$$\phi_{jk}^+(\gamma_j) + \bar{g}_j + \epsilon_k^h \le r_k \tag{7.115}$$

Then Alg. 8 always succeeds and outputs a set of estimation $\{\hat{h}_j\}$ with guarantees:

$$I_j(\hat{h}_j) = 1 \quad \wedge \quad \mathrm{dist}(\hat{h}_j, h_j^*) \le \gamma_j \tag{7.116}$$

Proof If $\epsilon_{\text{root}}^h \le \alpha_{\text{root}}$, there exists at least one sample point h_{root}^i so that $\mathrm{dist}(h_{\text{root}}^i, h_{\text{root}}^*) \le \alpha_{\text{root}}$ and $I_{\text{root}}(h_{\text{root}}^i) = 1$. Then the search over samples can find it.

Suppose for any vertex $j \in [t]$, Eqn. 7.116 holds. We now show for any $k \in [t+1]$, this

inequality still holds. For any $k \in [t+1]$, we have multiple answers $\hat{h}_{l\to k}$ from each of its parent l, by Thm. 7.4.1 and the fact that $\gamma_k < r_k$:

$$\mathrm{dist}(\hat{h}_{l\to k}, h_k^*) \leq r_k \tag{7.117}$$

and $I_k(\hat{h}_{l\to k}) = 1$. By Thm. 7.3.7, we can shrink the bound:

$$\mathrm{dist}(\hat{h}_{l\to k}, h_k^*) \leq \lambda_k \tag{7.118}$$

Therefore, by triangle inequality, the consistency check for vertex k will always be satisfied. Furthermore, a random pick of $\hat{h}_k = \hat{h}_{l^*\to k}$ satisfies Eqn. 7.116. ∎

In particular, for each leaf vertex k, we have:

$$\|\hat{\mathbf{p}}(k) - \mathbf{p}(k)\| = \|\hat{\mathbf{u}}_k - \mathbf{u}_k^*\| = \mathrm{dist}(\hat{h}_k, \hat{h}_k^*) \leq \gamma_k \tag{7.119}$$

and Alg. 8 successfully recovers the generating parameters $\mathbf{p}$.

Note that the condition $\mathrm{dist}(\tilde{h}_{\text{root}}, h_{\text{root}}^*) \leq r_{\text{root}}/2$ is not needed, since according to the sampling strategy, for any h_{root}^* there will be a sample sufficiently close to it and shows positive response. However, in this case, one needs to check false positives that may have a positive response on the top-most layer.

7.4.2 Sample and Time Complexity

The sample complexity of Alg. 8 can be obtained by simply counting the number of enumerations. To build the representations, the number of operations $N_{bottom-up}$ needed is:

$$Spent_{bottom-up} = \sum_j \#\{\mathbf{u}_j^i\} \cdots \#\{g_j^i\} \cdots |\mathcal{Z}_j| \cdot \left(\sum_{k \in ch(j)} |\mathcal{Z}_k| (\epsilon_k^h)^2 \right) \tag{7.120}$$

where $(\epsilon_k^h)^2$ is for local searching a compatible solution. On the other hand, the number of operations $N_{top-down}$ used to retrieve the estimation is:

$$Spent_{top-down} = \sum_j \sum_{k \in ch(j)} |\mathcal{Z}_k| \tag{7.121}$$

Therefore, the total time complexity is dominant by $Spent_{bottom-up}$, which is:

$$Spent = \sum_j \left(\frac{mn}{(\epsilon_j^h)^2} \right) \left(\frac{\bar{g}_j}{\epsilon_j^g} \right)^{2ch(j)} |\mathcal{Z}_j| \left(\sum_{k \in ch(j)} |\mathcal{Z}_k| (\epsilon_k^h)^2 \right) \tag{7.122}$$

The sample complexity N is similar:

$$N = \sum_j \frac{mn}{(\epsilon_j^h)^2} \left(\frac{\bar{g}_j}{\epsilon_j^g} \right)^{2|ch(j)|} |\mathcal{Z}_j| \tag{7.123}$$

where m and n are image dimension. Given a constant topology of hierarchical structure, the exponent of sample complexity N (as well as the time complexity) is upper-bounded by the number of fan-outs $\max_j |ch(j)|$ and is independent of the total dimensionality d of the deformation. In this sense, this algorithm breaks the curse of dimensionality!

However, such a big gain is not free at all. To make Alg. 8 works, one has to make sure that both conditions $\epsilon_{\text{root}}^h \leq \alpha_{\text{root}}$ and $\phi_{jk}^+(\gamma_j) + \bar{g}_j + \epsilon_k^h \leq r_k$ will be satisfied. The first condition forces us to sample densely, while the second condition indicates that we need to have small $\bar{g}_j$, a slow growing $\phi_{jk}^+(\cdot)$ and a large enough effective radius r_k. This leads to many trade-offs, for example, introducing type variables will reduce $\bar{g}_j$ since the children's degrees of freedom is partitioned, but will increase $\phi_{jk}^+(\cdot)$.

A more detailed discussion is in Sec. 7.5.

7.5 Discussion of the Algorithm

To achieve the guarantees, from bottom to top, following Eqn. 7.68, Eqn. 7.69 and Eqn. 7.70, α_j and r_j must decrease and γ_j must increase, as shown in Fig. 7.5. Similar to Top-down Hierarchical Models, the condition $\epsilon_{\text{root}}^h \leq \alpha_{\text{root}}$ indicates that α_j is the upperbound for sampling error and lower α_j means higher sample complexity.

On the other hand, γ_j, together with r_j, controls the usability of the algorithm. Since the condition (Eqn. 7.115):

$$\phi_{jk}^+(\gamma_j) + \bar{g}_j + \epsilon_k^h \leq r_k \tag{7.124}$$

has to be satisfied for all parent-child pair (j, k), we need to control the growth of γ_j and shrinkage of r_k over all the layers, especially among the top layers. On the other hand, the following condition (Eqn. 7.114)

$$\epsilon_{\text{root}}^h \leq \alpha_{\text{root}} \tag{7.125}$$

tells that the inverse sample density ϵ has to be smaller than α_{root}. Therefore, we need to control the shrinkage of α_j from bottom to top so that the sample density is kept low.

A few factors determine whether or not these two equations can be satisfied. The introduction of z_j for each patch j, is to squeeze the space of $\mathcal{G}_j$ and reduce $\bar{g}_j$. A small $\bar{g}_j$ is critical to control the trend of γ_j and r_j so as to make Eqn. 7.124 hold. A small $\bar{g}_j$ will also reduces the shrinkage of α_j. Increasing the inverse sample density ϵ_j^h and ϵ_j^g also help make the two equations hold. As a trade-off, both introduction of z_j and increasing ϵ will lead to higher computational burden.

Choice of the anchor point k for each parent vertex j is also an important factor. From the updating equations (Eqn. 7.68, Eqn. 7.69 and Eqn. 7.70), γ_j and r_j are determined exclusively by the anchor point $ah(j)$. Choosing the anchor point that has the largest discriminative range r_k will increase the discriminative range r_j for parent j. Similarly, choosing the anchor point with the smallest gray range γ_k will decrease γ_j.

Given a set of training images with labeled landmarks, it is thus important to determine how to build the initial map (i.e., function ***InitRep***) for better $(\alpha_k, \gamma_k, r_k)$, and at each layer which components are retained for further processing and which components are discarded. For the retained part, one needs to determine the size of $|\mathcal{Z}_j|$, and find good anchor points for each j. For the discarded part, one needs to find the exact space of $\mathcal{G}_j$ (and corresponding $\bar{g}_j$) to search from. If there exists such an appropriate assignment of $(\mathcal{H}_j, \mathcal{G}_j)$, then by Thm 7.4.2, we can get the globally optimal solution. This is the training stage of the algorithm.

7.6 Experiments on Synthesized Data

we have done preliminary experiments on synthesized data, which contains 300 training samples and 200 test samples and have been used in Top-Down Hierarchy. We use all test samples as performance evaluation. The Bottom-Up Predictions are implemented with C++/CUDA on NVidia Quadro 6000 GPU. It takes about 2-3 seconds to compute invariant representation of a test image and estimate the location of landmarks. In this experiment, there is only one type for every node j.

A few implementation tricks have been introduced to boost the performance. First, we use real value representations rather than binary representations. Second, given a parent state h_j and g_j, instead of checking $I_k(h_k)$ for only the compatible child states $h_k \sim (h_j, g_j)$, we also check $I_k(h'_k)$ for all h'_k that are close to h_k. This is essentially the same as having more children (and their representations) for a parent node. Since a parent node is activated only if all its child nodes are activated, this fine tuning will help eliminate more false peaks in the response map. We call this *child-expanding*.

Samples	50	100	200
Top-Down Hierarchical Prediction	5.11	4.78	4.55
Bottom-Up Hierarchical Prediction	4.63	4.07	3.93

Table 7.1: Performance comparison between Top-Down and Bottom-Up Hierarchical Predictions on Synthesized Dataset.

Given parent state, h'_k represents the compatible state of k's nearby (virtual) siblings. Therefore, although representation of the virtual node should also take high value (ideally 1 in the binary case), $I_k(h'_k)$ may not be high. Therefore, we need to match the score of $I_k(h'_k)$ between training and test. This involves storing response template during training and matching those templates during test. To reduce the dimensionality of the templates, we apply PCA and pick first 50 principle components. Every principle component is a 2D filter with small support. Therefore, the $\bigvee$ operation in Eqn. 7.40 essentially becomes 2D convolutions, and the template matching now becomes an inner product.

Furthermore, we can also collect templates that correspond to "negative" information. Ideally, without the child-expanding, the templates should be all-one vectors. This means that only if all the children are activated, the parent will be activated. With child-expanding, the templates may have entries that are lower than 1, but still they represent the child patterns that will activate the parent. Similarly, we can also store the child patterns that will *deactivate* the parent. Theoretically this is not necessary given sufficient number of samples, but practically this helps the performance.

Tbl. 7.1 shows the quantitative comparison between the Top-Down and Bottom-Up hierarchies. With the same number of training samples, the Bottom-Up Hierarchical Prediction performs better.

Chapter 8

Conceptual Comparisons between different methods

8.1 Data-Driven Descent

As mentioned before, our method is conceptually different from many existing methods. In the following, we describe this difference in a case-by-case study. To make the comparison and illustrations clear, we assume one-dimensional parameter space. In such a case, all distorted images generated from the distortion model form a one-dimensional manifold $\mathcal{I}$ (Eqn. (3.2)), shown as a curve in the image space (Fig. 8.1(a)). The template ($\mathbf{p} = 0$), the training samples and the distorted test image $I_{\mathbf{p}}$ are identified as points on the curve.

Generative/discriminative approaches. Fig. 8.1 shows the fundamental difference between our approach and generative and discriminative approaches in the image space. Generative approaches initialized from the template ($\tilde{\mathbf{p}} = 0$) converge to local optimum due to the complicated nonlinear structure of the manifold $\mathcal{I}$, as shown in Fig. 8.1(b). On the other hand, discriminative approaches can get the global optimum given the condition that the training samples densely cover the manifold $\mathcal{I}$, as shown in Fig. 8.1(c). This may not be a big deal if the manifold is one-dimensional, but will require enormous number of training samples in the high-dimensional case. Our approach achieves the same accuracy with an iterative strategy and much fewer training samples distributed in a radially decreasing way. The samples, especially those close to the origin, are heavily reused. While the maximum distance of two nearby training samples has to be $O(\epsilon)$-close in the discriminative case, the maximum distance between two training samples in our approach is only required to be smaller than the "gap" of the curve and *independent* of the prediction accuracy. The gap is implicitly encoded in the two universal constants L_1 and L_2 in

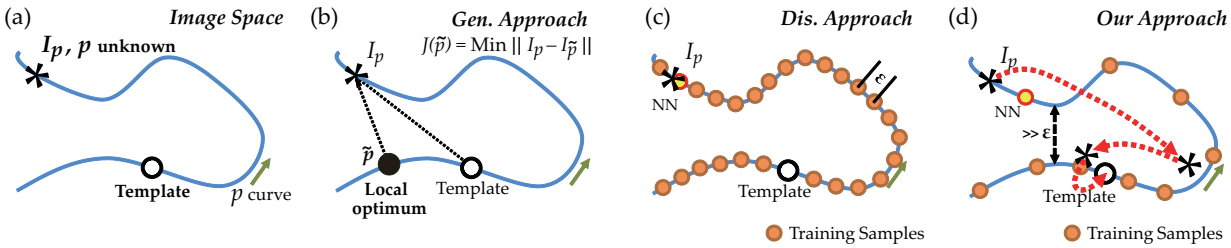


Figure 8.1: Comparison with generative/discriminative approaches, illustrated in the image space. **(a)** The image space. The curve parameterized by $\mathbf{p}$ is the set of all the distorted images $\mathcal{I}$ generated from the distortion model (Eqn. (3.2)), assuming one-dimensional parameter space. **(b)** Generative approaches initialized at the template ($\tilde{\mathbf{p}} = 0$) converge to the local optimum. **(c)** Discriminative approaches obtain an ϵ-accurate estimation, if the training samples densely cover the curve. **(d)** With much fewer samples than the discriminative approaches, our approach obtains the same accuracy by iteratively refining the parameter estimation, as illustrated by the dashed red arrows.

Eqn. (3.22).

Combining generative and discriminative approaches. Fig. 8.2(c)-(d) shows the difference between our method and previous methods combining the two approaches. Fig. 8.2(c) shows the heuristic that uses the discriminative approach as the initialization of the generative approach still leads to local minima, while our approach converges to the global optimum with the same distribution of training samples, as shown in Fig. 8.2(d). Although we do not guarantee global convergence with too few training samples, our approach fails only if the nearest-neighbor estimation is globally wrong, for example, predicting large negative values when the true parameter is large positive in 1-D case. In contrast, the way that the previous methods combine both approaches, as a generative approach by nature, is more sensitive to the local bumpy structures of the manifold $\mathcal{I}$.

Using warp-back strategies. Fig. 8.2(a)-(b) shows the fundamental difference between our approach and previous methods [5, 30, 102] with a similar strategy of successively warping-back. An energy minimization framework is commonly used in those methods. The standard gradient descent approach yields a trajectory of less distorted images until it reaches the template. By the formulation, the following two conditions have to be met: **(a)** the warp-back operations are in the warping family; **(b)** all the images on the trajectory have to be on the manifold $\mathcal{I}$, which is the set of all distorted images generated from the distortion model. This is only possible if the warping family forms a group.

For non-invertible distortion, if one condition is met then the other is broken. This is the reason why previous methods cannot handle non-invertible distortion as shown in Fig. 8.2(a).

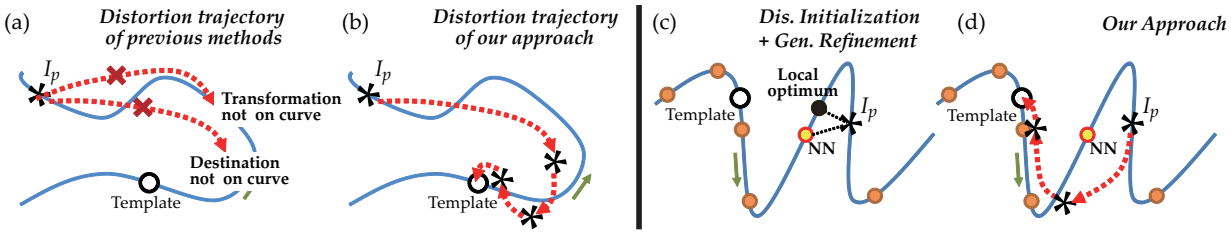


Figure 8.2: **Left:** Comparison with previous works [5, 30, 102] that also use warp-back strategy, illustrated in the image space. **(a)** Previous methods use a restricted formulation that requires both the intermediate distorted images *on* the curve and the warp-back distortions *in* the warping family, which is only possible for warping families that form a group. **(b)** Our approach allows the undistorted image *off the curve* during iterations and still achieves global convergence. **Right:** Comparison with other methods that combine the generative and discriminative approaches. **(c)** Using the discriminative approach to initialize the generative approach [75, 89] still leads to local convergence due to the local irregularity of the curve. **(d)** Using the same training set, our method converges to the global optimum.

However, our method can handle it by properly relaxing the condition (b) so that **(1)** the trajectory of less distorted images is allowed to be *off* the manifold yet **(2)** the trajectory converges to the manifold $\mathcal{I}$ when the parameter estimation is close to the true value, and is guaranteed to hit the template if the parameter estimation is perfect, as shown in Fig. 8.2(b).

Sample distribution. The convergence property of our algorithm is *independent* of the location of the test samples within the sphere $\|\mathbf{p}\| \leq r_0$, if the training samples are distributed as explained in Section 4.2.1. In other words, we attain the guarantee of the *worst-case* performance. This differs from many previous methods that only work for a given prior distribution. Furthermore, if the distribution of the parameters of real-world deformations of an object is known a priori, then it can be combined with our sampling strategy to reduce the number of training samples even further.

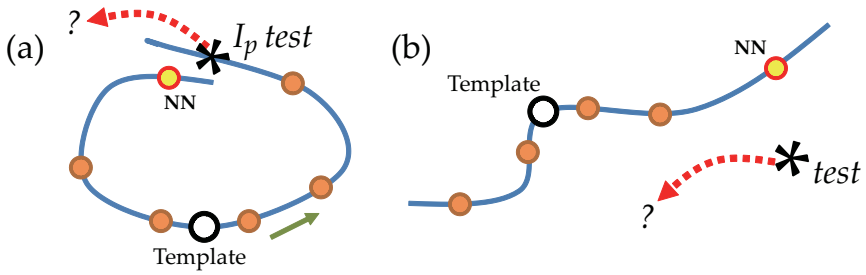


Figure 8.3: Two important failure cases. **(a)** One-to-many mapping case. The manifold $\mathcal{I}$ is (almost) self-intersecting. As a result, two similar images have very different parameters, one large positive and the other large negative. If we pull-back the test using the wrong parameter, then Algorithm 2 diverges. Note this does not violate Theorem 4.2.1 since in such cases, $L_2 \rightarrow +\infty$ and many more train samples are required especially near the ambiguous region to ensure each time the nearest-neighbor procedure picks the correct one. **(b)** The test distorted image is not on the curve. In such a case, the pull-back identity does not hold (Eqn. (3.9)). As a result, the image sequence of successive warping-back does not approach the manifold $\mathcal{I}$ and Algorithm 2 is not guaranteed to converge. This often happens in the case of occlusion, resampling artifacts or an incomplete distortion model. Yet we empirically show that in such cases, Algorithm 2 still gives decent results.

8.2 Bottom-up Hierarchy: Relation to Deep Learning and Graphical Models

The key components of bottom-up hierarchical methods is the following three equations to build invariant representations in a layer-by-layer fashion:

$$I_{k\rightarrow j}(h_j, g_j) = \max_{h_k \sim (h_j, g_j)} I_k(h_k) \quad (Translation) \tag{8.1}$$

$$\tilde{I}_j(h_j, g_j) = \bigvee_{k \in ch(j)} I_{k\rightarrow j}(h_j, g_j) \quad (Aggregation) \tag{8.2}$$

$$I_j(h_j) = \max_{g_j \in \mathcal{G}_j} \tilde{I}_j(h_j, g_j) \quad (Reduction) \tag{8.3}$$

8.2.1 Message Passing

The message passing updatings read as follows:

$$m_j(\mathbf{u}_j, z_j) = \sum_{k \in ch(j)} m_{k\rightarrow j}(\mathbf{u}_j, z_j) \tag{8.4}$$

$$m_{k\rightarrow j}(\mathbf{u}_j, z_j) = \min_{z_k} \left[\min_{\mathbf{u}_k} m_k(\mathbf{u}_k, z_k) + \theta^d_{jk}(\mathbf{u}_j, z_j, \mathbf{u}_k) \right] + \theta^c_{jk}(z_j, z_k) \tag{8.5}$$

We see message passing is just a soft version of our updating. First of all, the outgoing message $m_j(\mathbf{u}_j, z_j)$ is essentially the representation $I_j(h_j) = I_j(\mathbf{u}_j, z_j)$, while the incoming message $m_{k\to j}(\mathbf{u}_j, z_j)$ is $\max_{g_j \in \mathcal{G}_j} I_{k\to j}(\mathbf{u}_j, z_j, g_j)$. The only difference is:

(1) the order of aggregation and reduction is swapped.

(2) $\bigvee$ is replaced with a summation (rather than minimal operator).

(3) All the constraints appeared in Eqn. 7.39 now become soft penalties on objective function with proper weights (or Lagrangian multipliers).

8.2.2 Nonlinearity

In our framework, the operator $\bigvee$ is chosen as the minimal operator. This means for part j to be activated, all of its children have to be activated. This may fail in the presence of occlusion, where a few parts may be missing. On the other hand, message passing use $\bigvee = \sum$.

We can extend our framework so that $\bigvee$ can be a nonlinear function f that takes a weighted combination of $I_{k\to j}(h_j, g_j)$:

$$\tilde{I}_j(h_j, g_j) = f\left(b_j + \sum_{k \in ch(j)} w_{jk} I_{k\to j}(h_j, g_j)\right) \tag{8.6}$$

which coincides with deep learning framework. Using different b_j, w_{jk} and f yields different algorithms. More importantly, from the training data we can determine which parts are more stable and which parts are versatile, and set b_j and w_{jk} accordingly.

8.2.3 Criteria used in Unsupervised Deep Learning

Sparsity and reconstruction error are two major criteria extensively used to train deep architectures in an unsupervised and layer-by-layer manner. Although many papers have empirically shown their powers, few of them explain why they help. Indeed, if these criteria are not related to labels in supervised learning, how can they help in most supervised tasks?

It turns out that our bottom-up model can explain their effectiveness in a principled manner. For example, AutoEncoder minimizes reconstruction error of the input signal with some forms of regularization, e.g., imposing a norm penalty or using a limited number of model parameters. After optimization, the resulting parameters give a lossy reconstruction of the input signal, discarding information that has lower energy. To some extent, this coincides with our principle of lossy reconstruction. However, whether or not information with lower energy is related to the

high-level decision making remains unknown. If the answer is yes, then minimizing reconstruction error will give a good representation.

From our framework, we can clearly see that sparsity helps build a representation with less ambiguity (larger r_j) in Assumption 7.3.1. For example, in the lowest level, we can have multiple types for each part so that $I_k(\cdot, z_k)$ are sparser representations with larger "radius of acceptance" r_j, than its solo-type counterpart $I_k(\cdot) = \max_{z_k} I_k(\cdot, z_k)$. Similar things happen in higher levels by splitting response peaks into response maps of different types. Moreover, the sparser the responses are, the better different peaks are separated. The trade-off is that more response maps lead to higher computational cost.

Part III

Application-Specific Modeling

Chapter 9

Document Rectification

Over the past thirty years, optical character recognition (OCR) technology has matured to achieve very accurate results. Using OCR, printed books can be digitized rapidly into electronic form that can be easier to store, retrieve and edit. However, the document images input to OCR are required to be taken without distortion, i.e., the document must be planar with text lines being horizontal and straight. Any distortion significantly reduces the accuracy of OCR.

Traditionally the image of the document is acquired using a flat-bed scanner. While this is perfect for a single sheet of paper, forcibly flattening books (especially if they are old and precious) is not desirable. In order to address this problem, several vision systems estimate the distortion and rectify the image of the document. Some systems rely on additional and precisely calibrated hardware such as stereo cameras [103, 112], laser scanners [7], or structured light projectors [11, 12] to measure the 3D deformation in the documents. While these systems have demonstrated accurate results, they are more expensive and less portable and hence have not found widespread application. Other systems aim to reduce distortion by analyzing a single captured image of the document. The idea is to infer the distortions from the changes in scale and orientation of text lines and the foreshortening of text fonts. While these systems are cheap and flexible, estimation of the 3D deformation and rectification reliably from a single image is a challenging task.

In this paper, we follow the latter trend and build a vision system that reconstructs the 3D shape from a single image of curved document and rectifies the image (Fig. 9.1). We first estimate the 2D distortion (warping) grid in an image by exploiting the line structure and stroke statistics in text documents. This estimation consists of two main steps: text lines are automatically identified and densely traced, and the text orientation is determined at every location in the image. This approach does not rely on more noise sensitive operations such as image thresholding and character segmentation [13, 115], and does not rely on any *a priori* knowledge of the

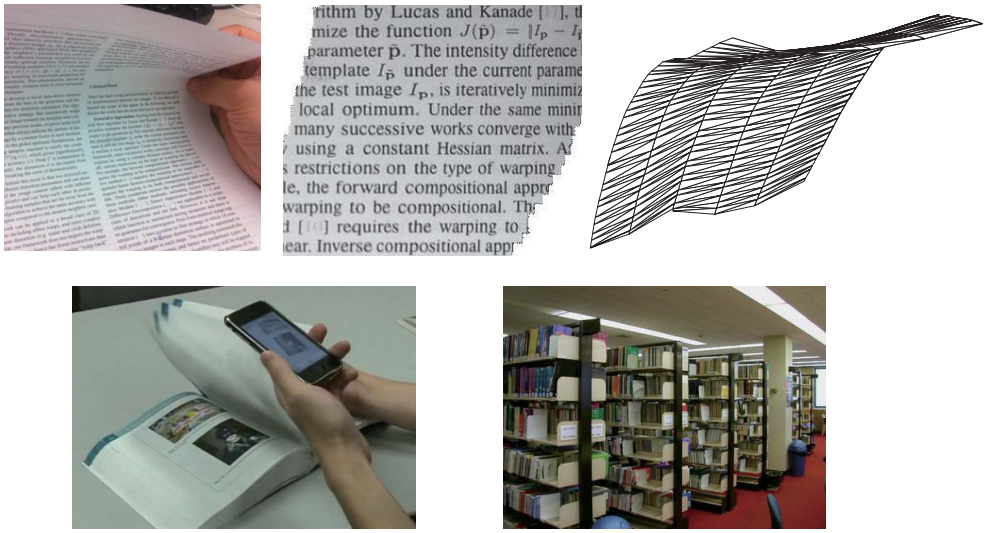


Figure 9.1: **First row:** From a single image of smoothly curved document as input, our methods compute the 3D shape of the document and a rectified image of text with no warping or shading effects. **Second row:** Two potential applications of our system: mobile text scanning and book digitization. (Images from Google).

font sizes, types or alphabet.

Unfortunately, knowing just the 2D distortion grid is not sufficient to rectify foreshortening and shading effects in the document. For this, we present a novel formulation of shape-from-texture to estimate the 3D deformation from the 2D distortion grid. In most documents, we observe that the 2D image grid can be regarded as a perspective projection of a *3D parallelogram mesh*. This observation allows us to solve an otherwise under-constrained reconstruction problem exactly using Singular Value Decomposition (SVD) (up to a global scale). Our reconstruction approach can be applied to general smooth surfaces and not restricted to simple parametric surfaces such as cylinders or developable surfaces as in [13, 50, 51, 117, 118]. Using the 3D shape, we present algorithms to unwarp the text document and remove shading effects under general and unknown lighting conditions.

Our system assumes that the image contains only text of the same font type and size. As shown in Fig. 9.1, our system has many applications. One example is mobile-OCR. Many smartphones have high-resolution cameras and can be used to image documents anywhere. Ideally, one just takes pictures of a note, notice, bulletin, receipt, or a book page and the application automatically converts the image to text. Another example is high-speed book scanning. In this scenario, a high-speed camera is used to record a book whose pages are being flipped through, and then the recorded video frames are rectified and assembled to obtain the textual content. This reduces the scanning time dramatically and can strongly impact several digital library initiatives.

9.1 Related work

Estimation of 2D document warping. Several approaches preprocess images using techniques such as binarization [13], connected component analysis [46, 117] or character segmentation [115] to estimate 2D warping. Previous line tracing techniques require pre-segmentation of each character [115], a global text line shape model (e.g. cubic spline [21]), or manual input of starting points of text lines [111]. These methods may miss many lines in the document. In contrast, we estimate the 2D distortions using domain knowledge in the form of the line structure and stroke statistics that is common to most text documents. Our self-similarity measure, scale estimation and line resampling steps do not miss lines, do not rely on thresholding and segmenting or identifying specific languages and fonts, and works even on low-resolution images.

3D reconstruction. To make 2D warp estimation more stable, many previous works assume a strong shape model, e.g., part of a cylinder [13], piecewise cylinder [118] or a developable surface [50, 51, 117]. In this paper, since the usage of domain knowledge leads to a better estimation of text warping, fewer assumptions are needed to reconstruct a broader class of 3D shapes. Most previous shape-from-texture works start by estimating the local differential quantities that the 3D shape projects onto the captured image, e.g. projected tangent angle [110], texture distortion [55] and foreshortening [25]. Since they all minimize non-linear objective functions, the estimation is not guaranteed to produce the global optimum [25]. In contrast, we formulate shape-from-texture in the specific context of text document images as a homogeneous least square problem, in which the globally optimal solution can be obtained using SVD.

9.2 Estimation of document image warping

We define warping in a document image as a two dimensional coordinate system with one coordinate along the text line direction and the other across the text lines. For convenience, we call the former *horizontal* lines and the latter *vertical* directions. In this section, we present a series of steps to accurately trace and identify the text lines using a self-similarity measure that works for different sizes and types of fonts and different alphabets. Next, we estimate the vertical directions (or text orientation) by exploiting local stroke statistics in the text. Compared to previous works [13, 50, 51], our methods use explicit domain knowledge to better estimate the two dimensional warping in document images.

9.2.1 Horizontal text line detection

We begin by tracing an initial set of text lines, called *seed lines*, across the document image from randomly selected starting points, based on an image self-similarity measure. Then these seed lines are resampled and refined using dynamic programming. We describe each of the steps below.

Line tracing using self-similarity measure

Fig. 9.3 illustrates the concept of self-similarity measure: the patches extracted from a set of points along a text line are similar to each other in terms of an image metric such as normalized correlation. This property holds for different languages, font types/sizes, illumination and resolution of document images, and thus can be used for line tracing. Unlike the procedure in [111], our measure is invariant to the choice of the starting point for tracing lines.

But how do we determine the scale (or size) of the patch in self-similarity measure? For this, we study how the *mean gradient magnitude* (MGM) changes over image scale. We compute an image pyramid by successively downsampling the original document image and for each level of the pyramid, we compute the MGM. We observe that the MGM initially increases during downsampling, since uniform 2D regions (inter-line whitespace) shrink more than 1D edges. However, the MGM starts to decrease at a scale where neighboring edges of letters/characters start to merge. This creates a peak as shown in Fig. 9.2(a). The location of the peak thus is directly related to the *characteristic scale* of the fonts in document images.

The text line tracing is done on the image downsampled to the characteristic scale. Starting from a random location x_0, we extract the patch centered at x_0, explore the patches at nearby locations $\{x_0 + (s\cos\theta_i, s\sin\theta_i)\}_{i=1}^{m}$ from x_0, and pick the one which is most similar to the current patch, measured by normalized correlation. Here s is the step and m is the number of angles to be explored. We repeat this process until the tracing trajectory reaches the boundary of the text region. We trace in both directions to cover the entire text region. The resulting lines are sorted from the top of the image to the bottom (Fig. 9.2(b)).

Resampling traced lines

Let $L \equiv \{l_1, l_2, \ldots, l_K\}$ be the *seed lines* traced and sorted as described above. Since the seed lines start from randomly selected points in the image, they typically skip text lines and may contain duplicate tracings for a single text line. A naive but inefficient approach would be to trace from every pixel of the image and pick a comprehensive set of text lines that cover the text region. Instead, using the fact that the directions of neighboring text lines are likely to be similar,

we can interpolate the sparse set L to obtain a dense set $L' \equiv \{\tilde{l}_1, \tilde{l}_2, \ldots, \tilde{l}_{K'}\}$ where $K' > K$.

From this dense set L', our goal is to pick exactly one tracing for each text line and inter-line whitespace. For this, we consider the *mean pixel intensity* (MPI) computed on each interpolated tracing $\tilde{l}$. $MPI(\tilde{l})$ is low on dark text lines and high on whitespaces. Therefore, from the top to the bottom of the document image, the MPIs of L' depicts a sinusoidal profile, and the local extremes of this profile (i.e., $MPI(\tilde{l}_i) > MPI(\tilde{l}_{i\pm1})$ or $MPI(\tilde{l}_i) < MPI(\tilde{l}_{i\pm1})$) yield the desired set of tracings, one for each text line and one for inter-line whitespace.

Line refinement

Each of the above set of tracings passes near the center of the text line or inter-line white space. However, this is not accurate enough for estimation of warping and rectification. To refine these tracings, we first identify the top and bottom of every text line by interpolation (as show in the rightmost figure of Fig. 9.4). This is because they are easier to localize than the line centers. Then we maximize the following objective function for the interpolated top or bottom tracing that is represented by a set of points $l = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$:

$$E(\delta y_1, \delta y_2, \ldots, \delta y_n) = \sum_{i=1}^{n} \phi_i(\delta y_i) + \lambda \sum_{i=1}^{n-1} \psi_{i,i+1}(\delta y_i; \delta y_{i+1}) \tag{9.1}$$

where $\{\delta y_i\}$ are the vertical shifts of the point (x_i, y_i). The first term $\phi_i(\delta y_i)$ measures the log-likelihood of a shifted point $(x_i, y_i + \delta y_i)$ being at the true top or bottom boundary of the text line. The second term $\psi_{i,i+1}(\delta y_i, \delta y_{i+1})$ is a smoothness measure that penalizes sharp changes in the tangents of the tracing. The shifts δy_i are bounded by adjacent text lines in order to avoid intersection of tracings.

Although the objective function is nonlinear, it can be solved exactly using dynamic programming in linear time. As shown in Fig. 9.2(d), the result of the above steps is an accurate identification and tracing of horizontal text lines.

9.2.2 Text orientation estimation using local stroke statistics

The alphabet in many languages, such as English, Chinese and Hindi, contain vertical strokes (e.g., “b”, “d”, “k” and “l” in English). This property can be exploited to estimate the vertical direction at each location of the text region. The vertical text direction along with the horizontal line tracing of the previous section constitutes the coordinate grid for the warping of the deformed document.

As shown in [50, 51], the stroke statistics can be captured by locating the peaks in the edge orientation histogram of a local region. However, it is nontrivial to find the right scale of such local regions. Small scales have good localization but can be unstable due to other interfering strokes in the letters, whereas a large scale is more stable but with poorer localization. Instead of simply smoothing over local estimations [51], we provide a formulation robust to interfering strokes and achieves stable estimation even in small scales.

Let Ω be the set of all the image pixels and $m(\mathbf{x})$ and $\theta(\mathbf{x})$ be the gradient magnitude and orientation at pixel $\mathbf{x}$. We partition Ω into M *overlapping* local regions $\{R_1, R_2, \ldots, R_M\}$. Our goal is to find $A \subseteq \Omega$ that ideally contains only the vertical strokes in the image, so that within each region R_i, the gradient orientations of A are similar. Once A is obtained, vertical direction can be estimated stably even in small scales. Mathematically, we optimize the following objective:

$$\begin{aligned} J(A) &= \sum_{i=1}^{M} J_i(A_i) = \sum_{i=1}^{M} J_i(A \cap R_i) \\ J_i(A_i) &= \sum_{\mathbf{x}\in A_i} m(\mathbf{x})(\theta(\mathbf{x}) - \bar{\theta}_{A_i})^2 - \beta \sum_{\mathbf{x}\in A_i} m(\mathbf{x}) \end{aligned}$$

where $A_i = A \cap R_i$ and $\bar{\theta}_{A_i} = \frac{\sum_{\mathbf{x}\in A_i} m(\mathbf{x})\theta(\mathbf{x})}{\sum_{\mathbf{x}\in A_i} m(\mathbf{x})}$ is the weighted average of local gradient orientations. The first term of $J_i(A_i)$ penalizes the weighted variance of gradient orientations of A_i. The second term is a regularization term to avoid the trivial solution $A = \emptyset$.

To solve this intractable combinatorial optimization, we introduce intermediate variables θ_i (the local dominant orientations) for each region R_i and write $J(A)$ as minimization of the following function $J'(A; \{\theta_i\})$ over $\{\theta_i\}$:

$$J'(A; \{\theta_i\}) = \sum_{i=1}^{M} J'_i(A_i; \theta_i)$$

where $J'_i(A_i; \theta_i) = \sum_{\mathbf{x}\in A_i} m(\mathbf{x})(\theta(\mathbf{x}) - \theta_i)^2 - \beta \sum_{\mathbf{x}\in A_i} m(\mathbf{x})$. Obviously $\min_A J(A)$ is equivalent to $\min_A \min_{\{\theta_i\}} J'(A, \{\theta_i\})$. We obtain a solution by alternatively minimizing θ_i and A_i for each region while fixing the region $A - A_i$ (A excludes A_i) and other variables θ_j ($j \neq i$). For each region R_i, we initialize θ_i as the perpendicular direction to the estimated horizontal text lines. An example result is shown in Fig. 9.5. The entire workflow of estimating the two dimensional warping of the document image is summarized in Fig. 9.4. We emphasize that accurate warping estimation is crucial for 3D reconstruction.

9.3 Reconstruction from a single image

Using the 2D warping, we can make the text line horizontal and text orientation vertical. However, this is not sufficient to rectify the document image due to the following two reasons. First, a pure geometric rectification cannot remove the shading on the images. Second, due to the depth variation, the foreshortening effects along the text lines cannot be correctly rectified as shown in Fig. 9.8(c). In this paper, we address these two problems by first estimating the 3D deformation of the curve document from only the 2D warping. Then the foreshortening effects can be rectified by using the depth variation along the text lines. The shading can be removed by computing surface normals of the 3D deformation and by assuming a reflectance model (e.g. Lambertian) for the document.

Without any assumptions, 3D reconstruction from a single image is an under-constrained problem with more unknown variables than constraints. In this work, we assume (1) the camera projection is perspective and (2) each cell of the 2D warping coordinate grid is a parallelogram in 3D space. The second assumption is reasonable because (a) the surface can be assumed to be locally planar or rigid if grid cells are sufficiently small, as demonstrated in recent work [94], and (b) for most *undistorted planar* documents, the text lines are parallel and so are local vertical text directions, thus forming a parallelogram grid.

But why not use a triangle mesh as in the work of Taylor et.al [94]? As shown in Fig. 9.6, the equilateral property in a parallelogram makes it possible to estimate its depth up to a global scale from a single perspective view. This is in contrast to [94] in which three camera views are required to reconstruct a 3D triangle up to a "flip" ambiguity.

We now formulate the problem of reconstructing a 3D parallelogram mesh from a 2D warping grid. Consider the illustration in Fig. 9.6. We denote the 3D coordinates of the i-th grid vertex as $\mathbf{V}_i = (X_i, Y_i, Z_i) = (x_i Z_i, y_i Z_i, Z_i)$, where (x_i, y_i) is its 2D coordinates. For simplicity, focal length is assumed to be 1 and center of projection is at the origin. Let $\{P_j\}_{j=1}^{N_p}$ denote the parallelograms where $P_{j,1:4}$ are the four vertices in counter-clockwise direction. The necessary and sufficient condition that the four vertices form a parallelogram is $\mathbf{\Delta}_{P_j} \equiv \mathbf{V}_{P_{j,1}} + \mathbf{V}_{P_{j,3}} - \mathbf{V}_{P_{j,2}} - \mathbf{V}_{P_{j,4}} = \mathbf{0}$. Thus we minimize the following objective:

$$Q(Z_1, Z_2, \ldots, Z_n) = \sum_{j=1}^{N_p} \|\mathbf{\Delta}_{P_j}\|^2 \tag{9.2}$$

To avoid the trivial solution of $\mathbf{Z} \equiv [Z_1, Z_2, \ldots, Z_n] = \mathbf{0}$, we add a global scale constraint $\|\mathbf{Z}\| = 1$ and solve Eqn. 9.2 exactly using Singular Value Decomposition (SVD) up to a global scale factor. Each 3D parallelogram brings forward 3 linear constraints, making the problem

Noise level	0	0.001	0.005	0.01	0.05
Errors	0.0012	0.0014	0.0044	0.0085	0.0503

Table 9.1: Reconstruction errors of randomly generated synthetic 3D shapes under different noise levels. Gaussian noise is added to the 2D projections, with standard deviations as shown. The average side length of grid cells is 1. The relative root-mean-square errors between ground truth and reconstructed 3D shapes are averaged over 100 random shapes for each noise level. The low errors demonstrate the robustness of our approach.

well-constrained.

For robustness to noise in the 2D grid locations, we add the re-projection errors to relax Eqn. 9.2:

$$Q'(\{\mathbf{V}_i\}_{i=1}^n) = \sum_{j=1}^{N_p} \|\mathbf{\Delta}_{P_j}\|^2 + \alpha \sum_{i=1}^{n} (X_i - x_i Z_i)^2 + (Y_i - y_i Z_i)^2 \tag{9.3}$$

where $\mathbf{V}_i = (X_i, Y_i, Z_i)$ is the estimated 3D location of the i-th grid vertex and α is the regularization constant. Eqn. 9.3 can also be solved exactly using SVD. Note that globally optimal solution is attained without initial guess of $\mathbf{Z}$.

There are a few special cases, e.g. plane and cylinder, in which the local parallelogram assumption is strictly true. In general, even if this assumption is only approximately true (because of local curvature), minimizing Eqn. 9.2 (or Eqn. 9.3) still gives very good estimations of a broad class of 3D shapes, including many non-ruled surfaces, as shown in Fig. 9.7. Fig. 9.9 shows a synthetic example in which text is mapped onto a sphere and projected back to the image plane. Using the methods in Section 9.2.1, we build the 2D coordinate grid and apply Eqn. 9.3 to obtain the 3D reconstruction. Note that, in principle, such surfaces cannot be reconstructed by previous approaches [50, 51, 117].

We quantitatively evaluate the 3D reconstructions obtained on a set of smooth surfaces randomly generated using 20 radial basis functions. Table 9.1 shows the relative root-mean-square errors between the ground truth and the 3D shape estimations. Gaussian noise is added to the 2D projections, with standard deviations as shown in the table. The average side length of grid cells is set to 1. The low errors demonstrate the robustness of our approach.

Note that more constraints could be incorporated into the optimization framework. A typical example is to enforce grid cells to be not only parallelograms but rectangles (text lines are horizontal and text orientation is vertical). However, such constraints introduce nonlinear terms in the optimization and global optimality can no longer be guaranteed.

9.4 Image rectification

9.4.1 Geometric rectification

While the vertical coordinates of the 2D warping grid is well-defined by interleaving text lines and white spaces, the horizontal coordinates along the text lines are not well-defined without depth information. An easy way to build the horizontal coordinates is to sample along the text lines with uniform image distance. This is a perfectly valid sampling for 3D reconstruction, but causes foreshortening effects in rectified text. As shown in Fig. 9.8(c), while all the text lines are horizontal, regions in the left appear stretched while regions in the right appear squished.

Fortunately, this foreshortening can be rectified using the 3D shape without knowing the font sizes, types and alphabet. Consider a patch within an image grid cell P_i. First we compute the 3D lengths of the two sides a_i and b_i of the parallelogram P_i and their ratio $r_i = a_i/b_i$. Then we warp the patch in P_i from the original image to a rectangle R_i of the same aspect ratio r_i. This is done by estimating a perspective transform that maps 4 corners of parallelogram P_i to the 4 corners of R_i. This process is applied to each grid cell independently. The result is shown in Fig. 9.8(d).

9.4.2 Photometric rectification

Using the estimated 3D shape, we can also remove the shading effects on the document image without knowing the prevailing lighting conditions. By assuming a Lambertian reflectance model, the pixel brightness at $\mathbf{x}$ is:

$$I(\mathbf{x}) = \rho(\mathbf{x})(\mathbf{n}(\mathbf{x}) \cdot \mathbf{w}) + \rho(\mathbf{x})A \tag{9.4}$$

where $\mathbf{w}$ is the unknown direction of lighting, A is the unknown ambient light and $\rho(\mathbf{x})$ is the unknown albedo. The surface normal $\mathbf{n}(\mathbf{x})$ can be computed from the 3D shape. We will further assume that the whitespace between lines (detected as described in Section 9.2) has uniform albedo. Then, we can set up a linear system of equations for patches in the whitespace to estimate the light direction $\mathbf{w}$, the ambient light A and the whitespace albedo ρ_w. The shading of the entire document image can be removed by computing the albedo image $\rho(\mathbf{x}) = I(\mathbf{x})/(\mathbf{n}(\mathbf{x}) \cdot \mathbf{w} + A)$. An example result is shown in Fig. 9.8(f).

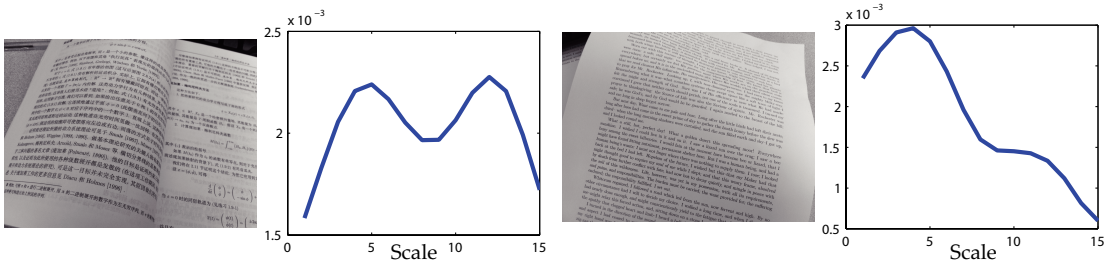
9.5 Experimental Results

We have applied our methods to documents with a wide variety of languages, font sizes and types, and challenging deformations. In order to demonstrate the ease of use, all the images were captured by an iPhone 4 camera. The focal length ($f = 2248$ pixels) is calibrated automatically within a few seconds using the *Theodolite app*. Fig. 9.12 shows representative results for curved documents written in English, Chinese and Hindi. From a single image, our system automatically reconstructs the 3D shape and rectifies the document given a few user input specifying the image region for line tracing. The third and fourth columns show accurate removal of both the foreshortening and the shading effects. Fig. 9.11 shows the histograms of the white spaces in the documents before and after photometric correction. The narrow peaks demonstrate the accuracy of our system.

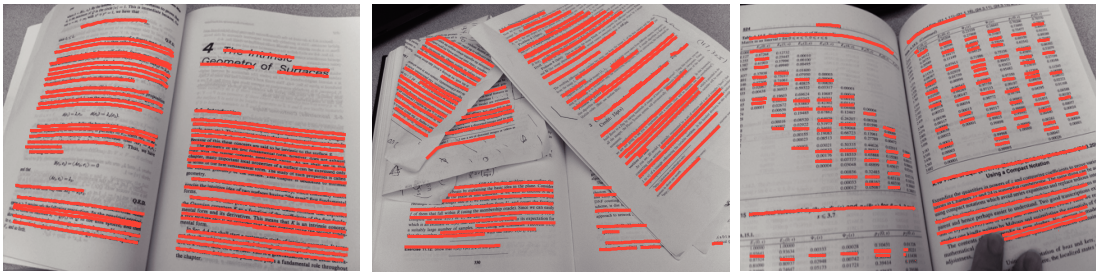
Failure cases. Fig. 9.10 shows several failure cases of line tracing. A large step (s large. See Section 9.2.1) in line tracing often yields line skipping, while a small step gives better results but runs slower. Besides, tracing is not working in non-text regions.

Performance. Our un-optimized MATLAB code takes 2-3 minutes to process an image (2592x1936) on Intel Core 2 (2.4GHz). The most time-consuming step is line refinement while others are fast. We are working on a C reimplementation on iPhone 4. The book imaging application requires fast capture but the processing can be done off-line.

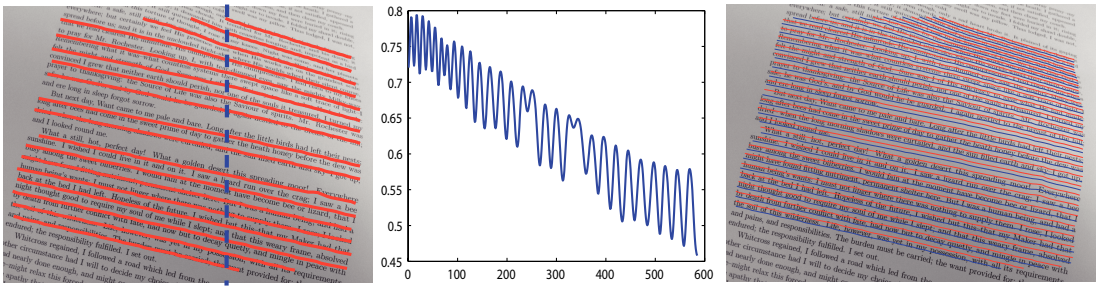
There are several avenues of future work. We wish to extend our system to handle more general documents with images, text, and illustrations and handle non-smooth deformations such as folds, creases and tears. We also wish to build a rapid book scanning system using a high-speed camera that captures the images of quickly flipping pages.



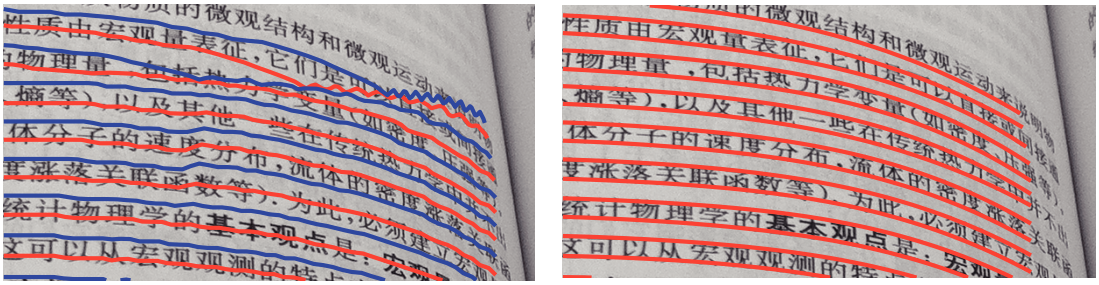
(a) Estimation of the characteristic scale



(b) Seed line tracing



(c) Line resampling



(d) Line refinement

Figure 9.2: Workflow of horizontal text line tracing. **(a)** The mean gradient magnitude (MGM) on each level of the image pyramid, computed by successively downsampling the document image. The first peak of MGM can be used as a characteristic scale of the text. **(b)** Line tracings from random starting points on document images. The tracing performs well in both text regions and white spaces. **(c)** *Left:* A set of tracings are chosen, called "seed lines"; *Middle:* Mean pixel intensities computed along densely interpolated seed lines. The centers of text lines and white spaces correspond to the local extremes of the mean pixel intensities; *Right:* Then the top and bottom of the text lines (blue and red) are estimated, **(d)** and are refined by optimizing Eqn. 9.1.

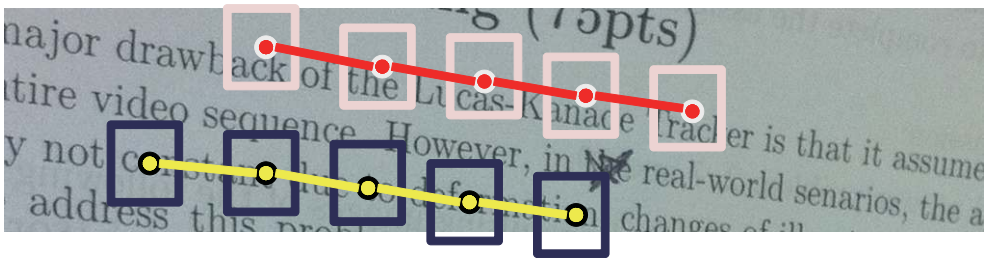
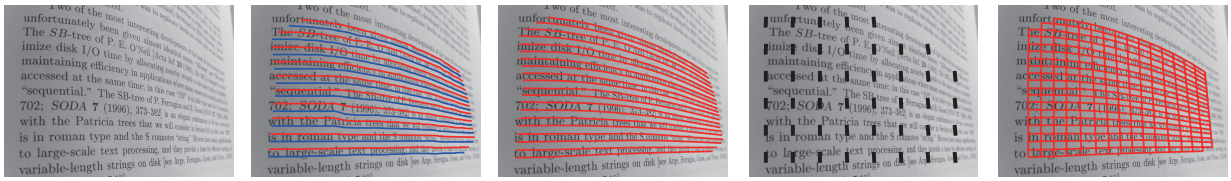


Figure 9.3: The self-similarity measure used for line tracing. Local patches extracted along the text line direction are correlated.



(a) Input image (b) Line tracing/resampling (c) Line refinement (d) Vertical estimation (e) Coordinate grid

Figure 9.4: Estimation of document image warping. **(a)** The original curved document image; **(b)** Horizontal text line tracing and resampling (Section 9.2.1-9.2.1); **(c)** Text line refinement (Section 9.2.1); **(d)** Estimation of vertical text orientation using local stroke statistics (Section 9.2.2); **(e)** The 2D coordinate grid of the image warp obtained using horizontal tracings and text orientation.

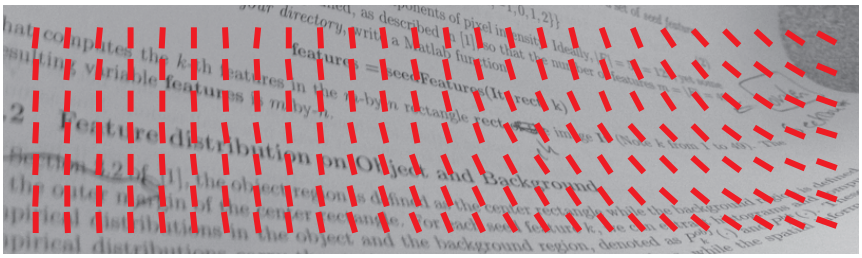


Figure 9.5: Example of text orientation estimation by Section 9.2.2.

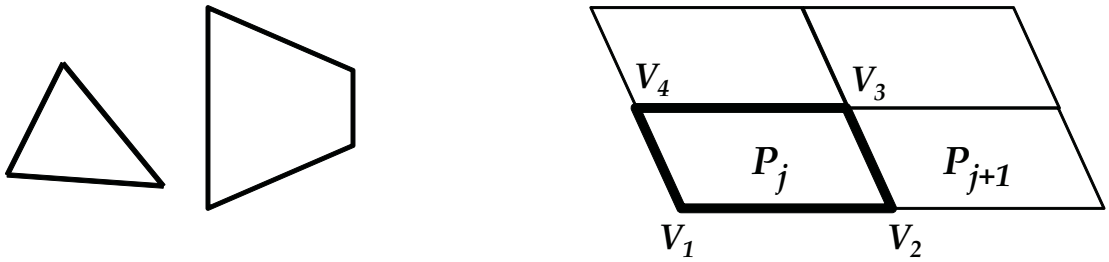


Figure 9.6: **Left:** Depth information can be extracted from a perspective projection of a 3D parallelogram using its foreshortened edges. This is impossible for a triangle. **Right:** We assume each grid cell is a parallelogram in 3D space, which gives 3 linear constraints: $\mathbf{V}_1 + \mathbf{V}_3 - \mathbf{V}_2 - \mathbf{V}_4 = 0$. With four unknowns, the parallelogram can be reconstructed up to a global scale. With more constraints than unknowns, estimating the depths of a grid with shared vertices is a well-defined problem.

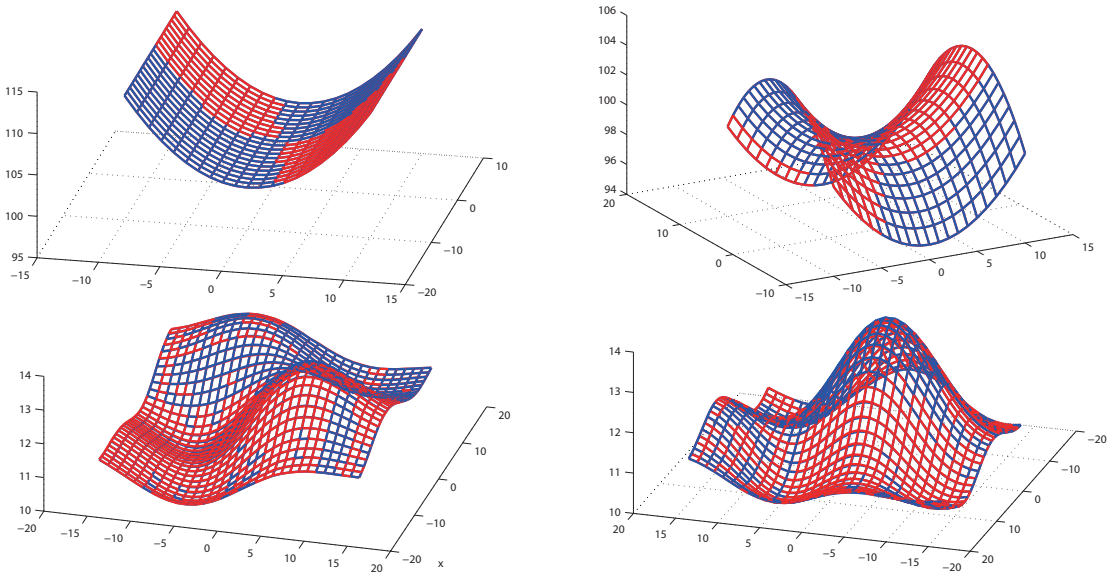
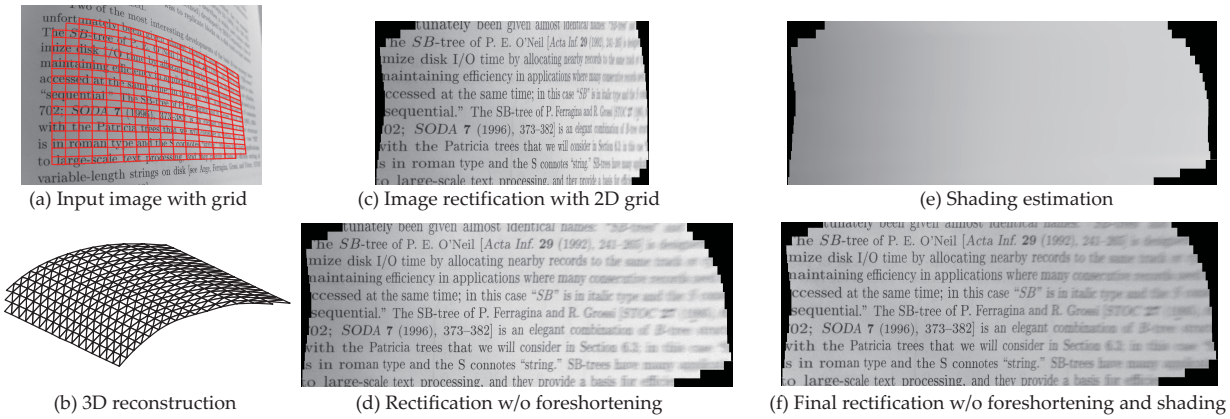


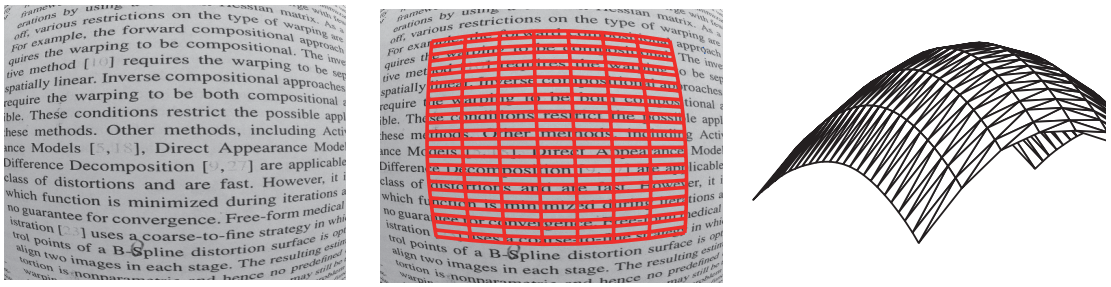
Figure 9.7: Example reconstructions of synthetic shapes. The ground truth shapes are shown in blue, while reconstructed shapes using Eqn. 9.3 are shown in red. Our method can reconstruct both ruled and non-ruled surfaces.



(a) Input image with grid (c) Image rectification with 2D grid (e) Shading estimation

(b) 3D reconstruction (d) Rectification w/o foreshortening (f) Final rectification w/o foreshortening and shading

Figure 9.8: 3D reconstruction and image rectification. **(a)** Original image with the 2D coordinate grid (Section 9.2); **(b)** 3D reconstruction from a single image (Section 9.3); **(c)** Image rectification using the 2D coordinate grid. Notice the foreshortening and shading effects. Using 3D information, **(d)** foreshortening can be rectified (Section 9.4.1) and by exploiting a reflectance model (e.g. Lambertian), **(e-f)** shading can be estimated and normalized to yield an albedo image. (Section 9.4.2).



(a) (b) (c)

Figure 9.9: Example 3D reconstruction of text printed on a sphere. **(a)** The input image; **(b)** The estimated 2D coordinate grid (Section 9.2); **(c)** 3D reconstruction using Eqn. 9.3.

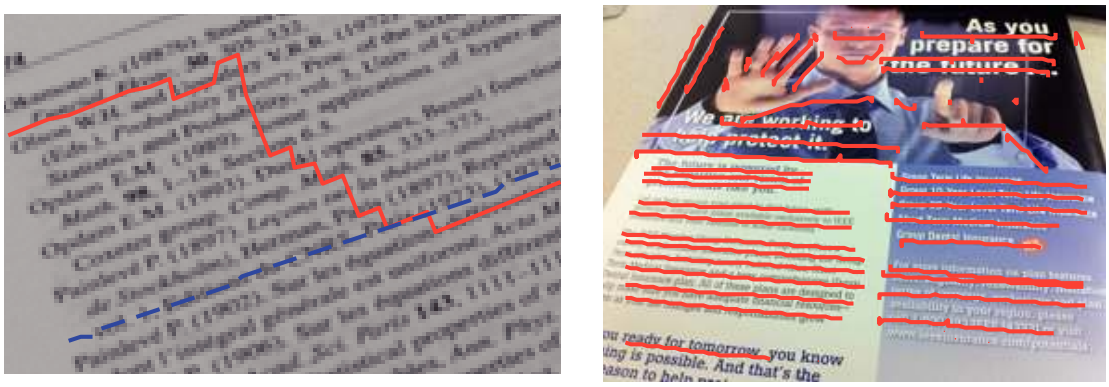


Figure 9.10: Failure cases in line tracing. **(a)** Tracing with large step ($s = 5$) yields line skipping (red solid line). Tracing with small step ($s = 3$) tends to suffer less from line skipping (blue dotted line). **(b)** Line tracing on complicated text layout with images. The algorithm tends to follow straight lines in non-text region.

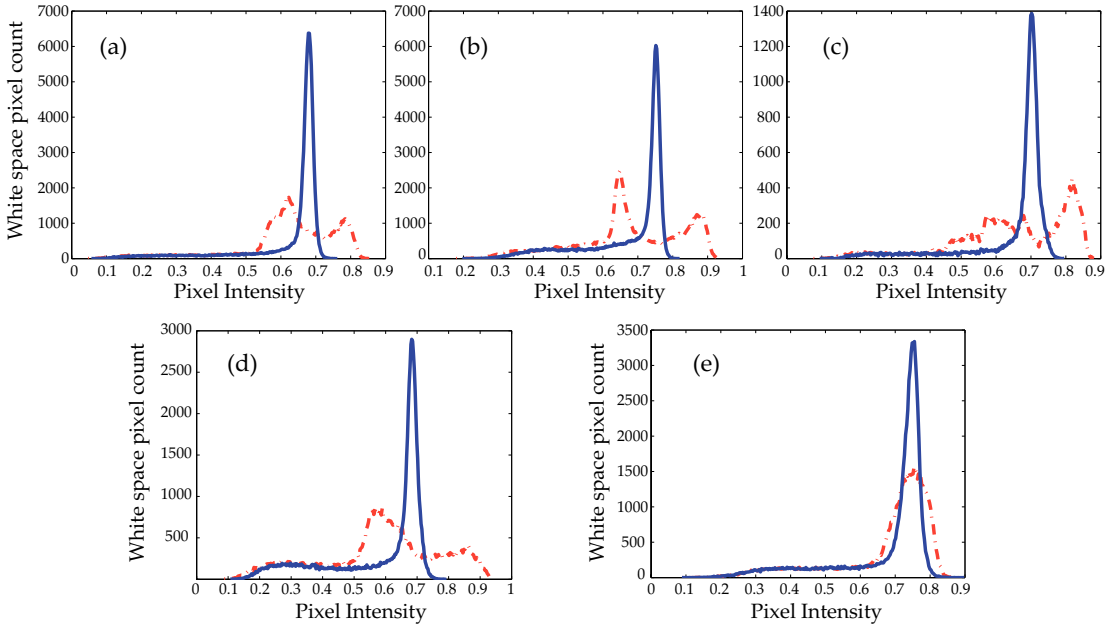


Figure 9.11: Histograms of white spaces in the documents before (red dotted lines) and after (blue solid lines) shading removal. The five histograms correspond to the five document images from top to bottom in Fig. 9.12. The intensity distributions on the computed albedo images are significantly narrower after shading removal.

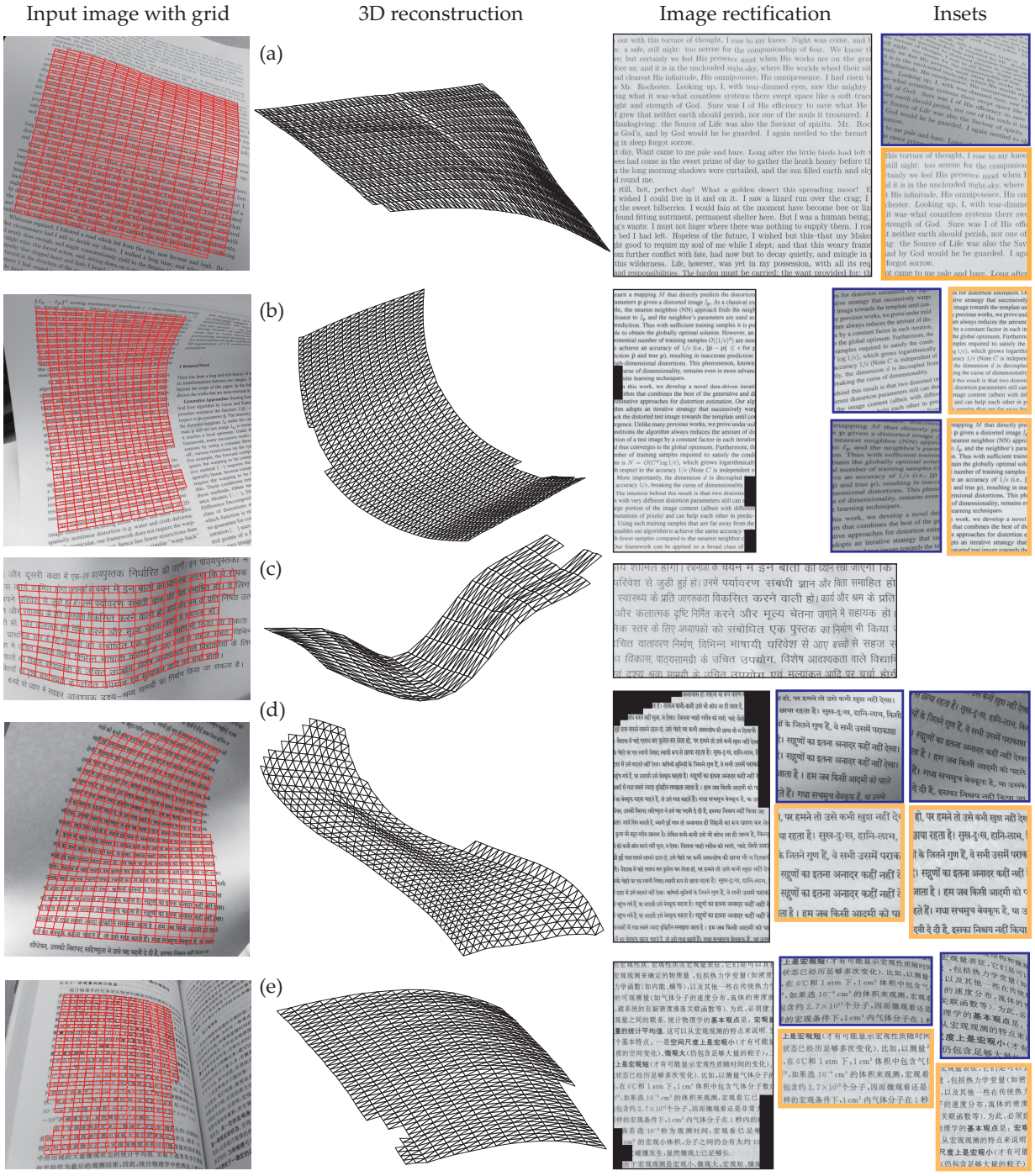


Figure 9.12: Image rectification and 3D reconstruction from a single curved document image. **First column:** Estimated 2D coordinate grid; **Second column:** 3D reconstruction. **Third column:** Image rectification. **Fourth column:** The insets show comparisons between rectified images (orange rectangles) and original distorted images (blue rectangles). The geometric deformations, text foreshortening and shading effects are all removed by our system. **(Please zoom in to see the details.)**

Chapter 10

Depth from Turbulence

The visual manifestations of clear air turbulence occur often in our daily lives — from hot kitchen appliances like toasters and ovens, to plumes of airplanes, to desert terrains, to roads on hot summer days, to the twinkling of stars at night. The shimmering and distortion observed are caused by random fluctuations of temperature gradients near warm surfaces. In this case, the image projection of a scene point viewed through turbulence is no longer a deterministic process, and often leads to poor image quality.

Several works in remote sensing and astronomical imaging have focused on image correction through turbulence. For atmospheric turbulence, the distorted wavefronts arriving from stars can be optically corrected using precisely controlled deforming mirror surfaces, beyond the angular resolution limit of telescopes [67]. For terrestrial imaging applications, recent works have proposed to digitally post-process the captured images to correct for distortions and to deblur images [28, 31, 35, 41, 122]. Optical flow based methods have been used further to register the image sequences to achieve modest super-resolution [84].

While previous works have focused on what turbulence does *to* vision, this article addresses the question of what turbulence can do *for* vision. In other words, what information about the scene can be extracted when viewed through turbulence? Based on the physical model of wave propagation, we study the relationship between the scene depth and the amount of distortion caused by homogeneous turbulence over time (see an intuitive illustration in Fig 10.1). Then, we extend this relationship to more practical scenarios of finite extent and height-varying turbulence, and show how and in what scenarios we can estimate depth ordering, depth discontinuity and relative depths. Although general non-homogeneous turbulence does not directly yield depth information, its statistical property can be used along with a stereo camera pair to improve long-range depth estimation.

The input to our techniques is a sequence of short exposure images captured from a stationary

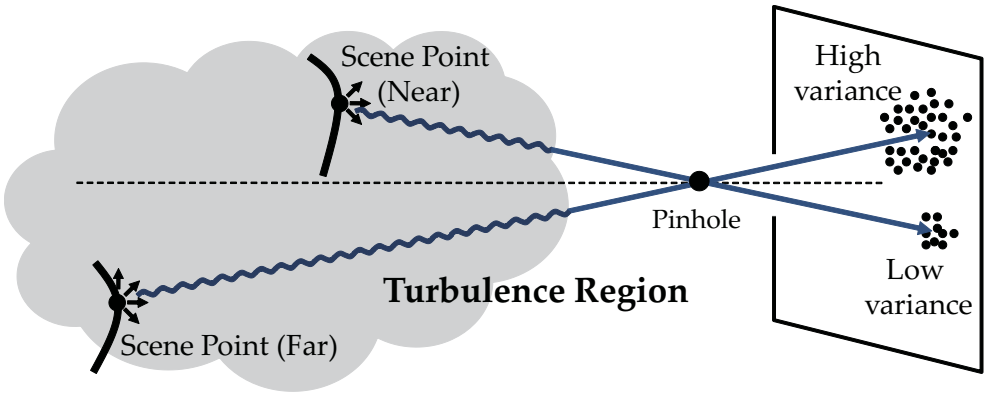


Figure 10.1: Random fluctuations in the refractive index of a medium cause the perturbation of a light wave radiating from a scene point. The resulting image projections of the scene point over time are also random. The longer the distance of a scene point from the camera, the greater the variance of its image projection.

camera (or camera pair). Depth cues are obtained by first tracking image features and then by computing the variances of tracker displacements over time. Any feature tracking algorithm can be applied, such as that based on template matching. We verify our approaches in both laboratory and outdoor settings by comparing against known (ground truth) distances of the scene from the camera. We also analyze how the depth cue estimation is influenced by the parameters of the imaging system, such as aperture, exposure time and the number of frames. The depth information computed is surprisingly accurate, even when the scene and camera are not within the turbulent region. Hence, we believe that turbulence should not be only viewed as "noise" that an imaging system must overcome, but also as an additional source of information about the scene that can be readily extracted[1].

10.0.1 Related Work

Characterizing the structure of turbulence is one of the open problems in physics, with a long research history, starting from the early methods of Kolmogorov [44]. For this work, we reference multiple textbooks by Kopeika [47], Tatarskii [93], Ishimaru [38] and Roggemann [67]. To our knowledge, the key physical model (Eqn. 10.3) in these texts has not been exploited by the computer vision community.

Direct measurement of turbulent media has received much attention in fluid dynamics. Shadowgraph and Schlieren imaging [79, 109] techniques are often used to capture the complex airflow around turbines, car engines and airplanes wings. Image displacement observed in turbulent

[1]While not the focus of this work, the short exposure (noisy) and distorted input images can be combined using a dense image alignment approach [99] to improve image quality.

Figure 10.2: Turbulence due to aircraft plumes or road surfaces on a hot day causes shimmering, distortion and blurring in images.

media has been shown to be proportional to the integral of the refractive index gradient field. This property is exploited in a tomographic approach [33] with many views to compute the density field of the medium from image displacements of known backgrounds. This approach, called Background Oriented Schlieren (BOS) imaging [65, 104, 105], has emerged as a new technique for flow visualization of density gradients in fluids. Such approaches have also been used to render refractive volumes of gas flow [4]. Similarly, there has been work [60] that aims to estimate the shape of a curvy refractive interface between two media (water and air, for example) using stereo and known backgrounds. In contrast, our work exploits image displacements to extract the depths cues of an unknown scene using an image sequence captured from a single viewpoint.

10.1 Characterization of Turbulence

Turbulence causes random fluctuations of the refractive index $n(\mathbf{r}, t)$ at each location $\mathbf{r}$ in the medium and at time t. From Kolmogorov's seminal work [44, 45], $n(\mathbf{r}, t)$ forms a random field in space-time and can be characterized by a structure function $D(\mathbf{r}_1, \mathbf{r}_2, t)$ that computes the expected squared difference of refractive index at two distinct spatial locations $\mathbf{r}_1$ and $\mathbf{r}_2$:

$$D(\mathbf{r}_1, \mathbf{r}_2, t) = \langle |n(\mathbf{r}_1, t) - n(\mathbf{r}_2, t)|^2 \rangle \,. \tag{10.1}$$

For *stationary* turbulence, the structure function is constant over t, i.e., $D(\mathbf{r}_1, \mathbf{r}_2, t) = D(\mathbf{r}_1, \mathbf{r}_2)$. Stationary turbulence is *homogeneous* if $D(\mathbf{r}_1, \mathbf{r}_2) = D(\mathbf{r})$, where $\mathbf{r} = \mathbf{r}_1 - \mathbf{r}_2$. This means that the structure function depends only on the relative displacement of the locations. Homogeneous turbulence is *isotropic* if the structure function is spherically symmetric, i.e., $D(\mathbf{r}) = D(r)$,

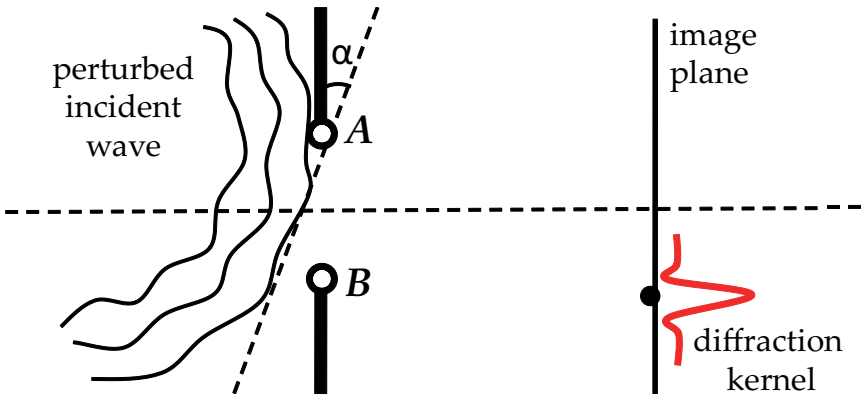


Figure 10.3: The phase difference of an incident wave (e.g., the phase of point B *leads* that of A) at the aperture determines the angle-of-arrival α (AoA) and in turn, the center of the diffraction kernel, i.e., the location of the projected scene point in the image plane.

where $r = \|\mathbf{r}\|$. From dimensional analysis, Kolmogorov shows that the structure function follows a $2/3$ power law [93]:

$$D(r) = C_n^2 r^{2/3}, \tag{10.2}$$

where, the constant C_n^2 reflects the strength of turbulence. For non-homogeneous turbulence, C_n^2 is a function of absolute location. A non-turbulent atmosphere has $C_n^2 = 0$.

In general, the strength C_n^2 of turbulence depends on a variety of environmental and physical factors, such as temperature, pressure, humidity and wavelength of light, which in turn depend on the time of day (less during sunset and sunrise, more at mid-day), cloud cover (less during cloudy day and more during cloudy nights), and wind patterns. An empirical relationship between these factors and refractive index changes can be found in Kopeika's textbook [47].

10.2 Image Formation through Turbulence

When an electromagnetic wave propagates through a turbulent medium, it undergoes random fluctuations in both amplitude and phase. The perturbed phase determines the angle-of-arrival (AoA) of the light incident at the camera, which in turn fixes the projected location of the scene point in the image (Fig. 10.3). Mathematically, the propagation of an electric field under the influence of the turbulence structure function in Eqn. 10.2 can be obtained by solving Maxwell's equations. Since most surfaces and sources of interest to computer vision are at finite distances from the camera and produce divergent waves, we will consider the propagation of spherical waves. Then, following the derivations in [38, 47], the variance $\langle \alpha^2 \rangle$ of the angles-of-arrival of the waves from a scene point at distance L from the camera is obtained by integrating along the

line of sight:

$$\langle \alpha^2 \rangle = 2.914 D^{-1/3} \int_0^L C_n^2(z) \left(\frac{z}{L}\right)^{5/3} \mathrm{d}z, \tag{10.3}$$

where, D is the diameter of the aperture. The actual fluctuation $\langle \delta^2 \rangle$ of the projected image location can be computed using the relation $\delta = f \tan \alpha$ where, f is the focal length of the camera. For small angles, $\delta \approx f\alpha$.

In the following, we will discuss three important special cases of the above image formation model. We will address the general case of non-homogeneous turbulence in Section 10.6. First, consider a scenario where both the camera and the scene of interest are immersed in a homogeneous turbulence medium (for example, a road scene with vehicles on a hot summer day), as illustrated in Fig. 10.4(a). Since C_n^2 is a constant, we can integrate Eqn. 10.3 to obtain:

$$\begin{aligned} \langle \alpha^2 \rangle &= 2.914 D^{-1/3} C_n^2 \int_0^L \left(\frac{z}{L}\right)^{5/3} \mathrm{d}z \\ &= \frac{3}{8} K_n^2 L, \end{aligned} \tag{10.4}$$

where, $K_n^2 = 2.914 D^{-1/3} C_n^2$. So, the variance of projected positions of the scene point in the image plane over time is directly proportional to the distance L between the scene point and the camera. Setting aside the issue of spatial resolution, this linear relationship determines depth with constant precision for all distances within the turbulence region. By comparison, in stereo, the depth precision falls as the square of the distance from the camera to the scene.

In many scenarios, like the plume of an aircraft or a steaming kettle, the source of turbulence may not extend over the entire line-of-sight from the camera to the scene. In this case, we will assume local homogeneity, i.e., C_n^2 is a constant within a short range and zero elsewhere. For convenience, we decompose L into three parts: $L = L_s + L_t + L_c$, as illustrated in Fig. 10.4. L_s is the distance between the scene point and the turbulence region, L_t is the path length within the turbulence region and L_c is the distance between the camera and the turbulence region. Once again, we can integrate Eqn. 10.3 to obtain the analytic form:

$$\begin{aligned} \langle \alpha^2 \rangle &= \frac{K_n^2}{L^{5/3}} \int_{L_s}^{L_t + L_s} z^{5/3} \mathrm{d}z \\ &= K_n^2 \frac{3}{8 L^{5/3}} \left((L_t + L_s)^{8/3} - L_s^{8/3} \right). \end{aligned} \tag{10.5}$$

Letting $\langle \alpha_{\text{relative}}^2 \rangle = \frac{3}{8L^{5/3}} \left((L_t + L_s)^{8/3} - L_s^{8/3} \right)$ allows us to write in short:

$$\langle \alpha^2 \rangle = K_n^2 \langle \alpha_{\text{relative}}^2 \rangle. \tag{10.6}$$

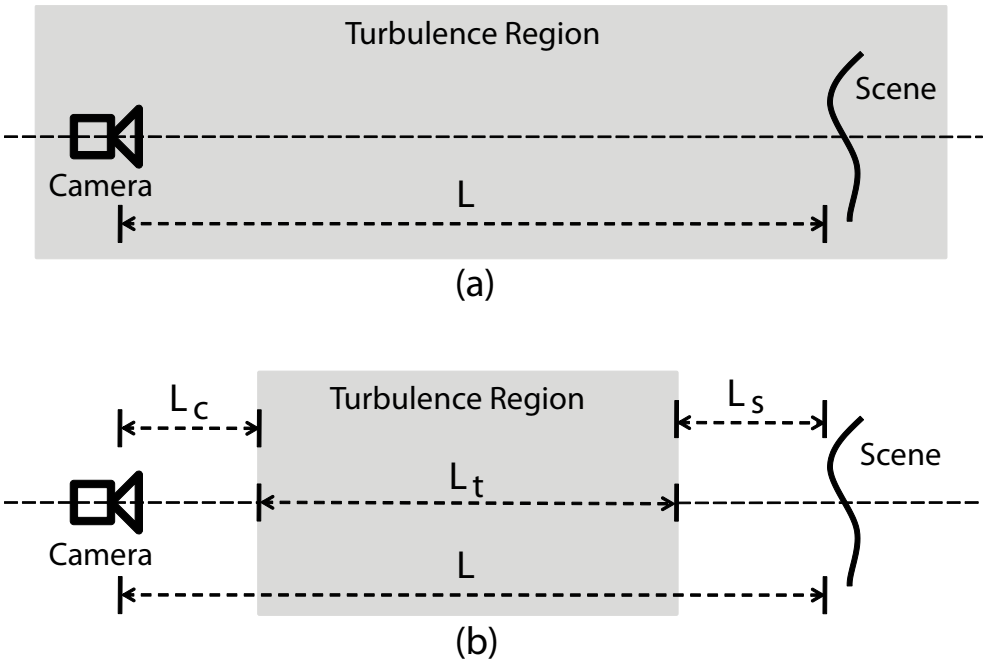


Figure 10.4: Image formation through turbulence. **(a):** Both the camera and the scene are immersed within a homogeneous turbulence region. **(b):** The camera and/or scene are outside the turbulence region.

If we fix the camera location L_c and the turbulence region L_t, and move the scene point away from the camera, the variance is a monotonically increasing function with respect to L, as shown in Fig. 10.5. From this, we observe that the variance increases even if the scene moves away from the turbulence region. This is a counter-intuitive result that cannot be explained by ray optics (hence, the usage of "waves" in this article). The variance, however, converges to a fixed value $\langle \alpha_\infty^2 \rangle$, when the scene point is infinitely far away from the camera (e.g., a distant star). This can be seen by taking the limit $L_s \to \infty$ in Eqn. 10.6 to obtain:

$$\begin{aligned} \langle \alpha_\infty^2 \rangle &= K_n^2 \lim_{L_s \to \infty} \frac{3}{8L^{5/3}} \left((L_t + L_s)^{8/3} - L_s^{8/3} \right) \\ &= K_n^2 L_t. \end{aligned} \tag{10.7}$$

In this case, the light emitted by the scene point can be modeled as a plane wave.

Height-varying turbulence. In practice, the air turbulence may not be homogeneous in the entire field of view. For example on an asphalted road, the turbulence is more severe near the road surface than away from it. We model this effect by writing the strength of turbulence as a

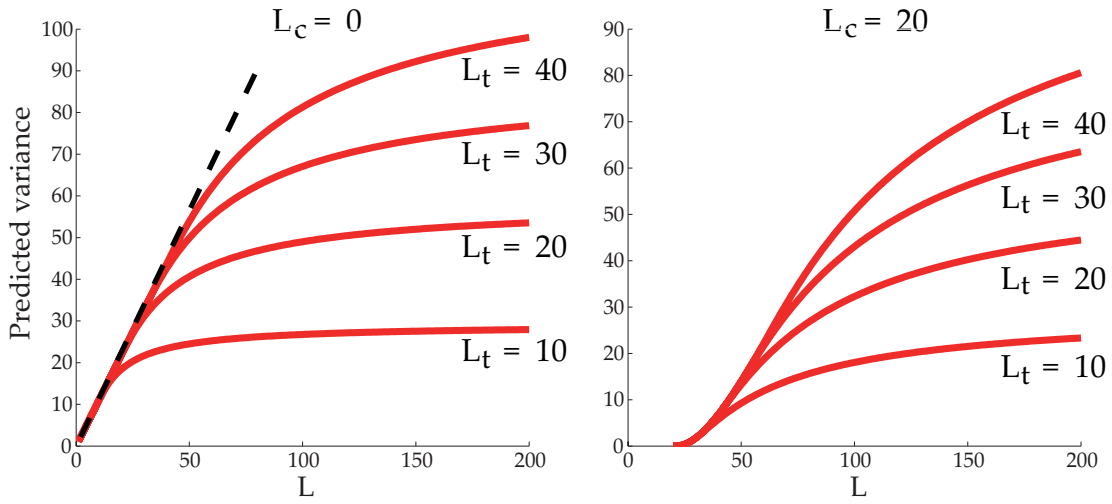


Figure 10.5: The variance of the angle-of-arrival predicted by Eqn. 10.6 under different experimental settings. For each curve, the camera and the extent of turbulence (L_c and L_t) are fixed, while the scene depth (L_s) is varied. Each curve represents a monotonically increasing function of scene depth that converges to a fixed variance (at infinity). The dashed black line is the linear relation in the special case when $L_c = L_s = 0$.

smoothly varying function of height h, yielding a separable model:

$$\langle \alpha^2 \rangle = K_n^2(h) \langle \alpha^2_{\text{relative}} \rangle. \tag{10.8}$$

Typically, $C_n^2(h)$ (or $K_n^2(h)$) decreases with respect to h.

10.3 Depth cues from an Image Sequence

In this section, we investigate what depth cues can be obtained from the observed variance of image displacements of scene points. The input to all our algorithms is a sequence of images of a stationary scene viewed through turbulence by a fixed camera. Once the images are captured, we track a sparse set of distinctive feature points on the scene. While any feature-tracking algorithm may be used, we adopt a simple frame-to-frame template matching approach. To handle image blurring caused by turbulence, we also add blurred versions of the templates. The variance of the image location of each tracked point is then computed.

For a fixed configuration of camera and extent of the homogeneous turbulence region, the model (Eqn. 10.6) is a monotonic smooth function of scene depth. Thus, both depth ordering and discontinuities (like two buildings far apart) of the scene can be readily obtained from variances. In particular, detecting such discontinuities can be useful to segment the scene into different

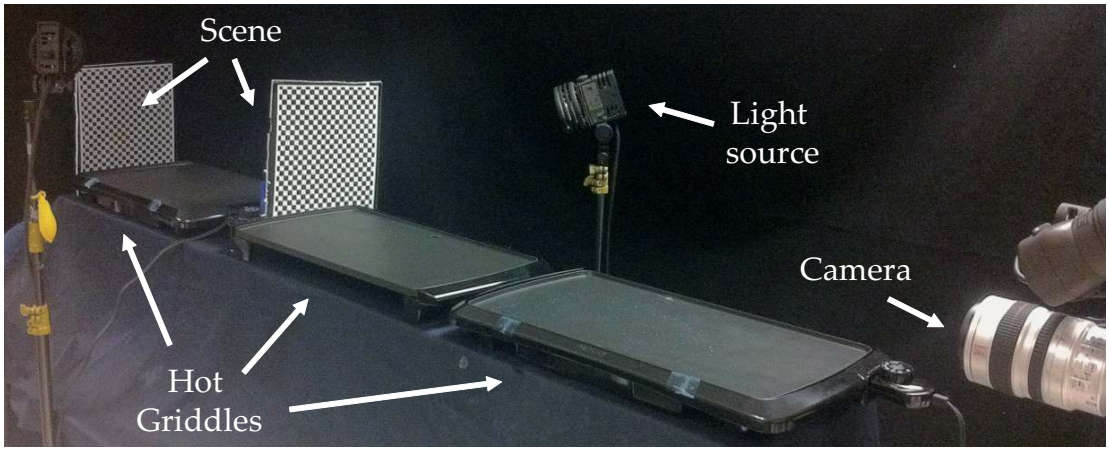


Figure 10.6: Experimental setup: Three adjacent electric cooking griddles are heated up to 400 degrees Fahrenheit to create hot air turbulence. A camera observes a scene through the turbulence. By varying the temperature, we can emulate a variety of outdoor turbulence strengths and path-lengths of several kilometers.

depth layers (planes).

On the other hand, a more quantitative measurement, such as relative depth between scene points, requires additional assumptions. Note that absolute depth cannot be computed without knowing the turbulence strength, C_n^2. Thus, without loss of generality, we will assume $L_t = 1$. When the camera and scene are immersed in turbulence ($L_c = L_s = 0$), the linear variance-depth relationship (Eqn. 10.4) allows us to obtain relative depth by taking variance ratios to eliminate the unknown constant K_n^2. In general, if L_c, L_s and K_n^2 are known, depth can be obtained by inverting Eqn. 10.6. By monotonicity of the model, only a unique depth can be obtained from a given variance.

However, in practice, these constants are usually unknown. Thus, for N scene points we have $N + 3$ unknowns (N depths plus L_c, L_s and K_n^2) but N equations. Consider a scene with repetitive patterns (windows on a building, street lamps, cars parked on a street), then the depths $\{L_i\}_{i=1}^N$ of the N points follow an arithmetic sequence:

$$L_i = L_0 + i\Delta L \tag{10.9}$$

Thus, N depths $\{L_i\}_{i=1}^N$ are parameterized by 2 variables, L_0 and ΔL. As a result, only $2+3 = 5$ scene points suffice to estimate (using numerical optimization) both the relative depths and the extent of the turbulence region.

In the case of height-varying turbulence, we need to estimate the height-varying function $K_n^2(h)$ as well as the scene depth. Fortunately, if the height is aligned with the y-axis of the

image, then separating depth from height can be achieved by treating each scan-line individually.

10.4 Laboratory Experiments

We performed several experiments in a controlled laboratory environment to validate the theory. A flat cooking griddle of size 52cm $\times$ 26cm is used to produce and maintain uniform heat of up to 400 degrees Fahrenheit, across the flat surface. In the experiment setup (Fig. 10.6), multiple such griddles are placed side by side to increase the path-length of turbulence. The three griddles set at maximum temperature produce roughly the same shimmering as a kilometer of natural turbulence in the desert. By controlling the number of griddles and the temperature, a wide range of turbulence strengths seen outdoors can be emulated. In all experiments, variances are estimated by capturing a 20-30 seconds long video sequence of the scene at 30 fps.

10.4.1 Quantitative Evaluation

Variance-depth linearity within turbulence region. 50 equally-spaced LEDs are placed 5 cm above the hot griddle (in the turbulence region). One end of the stick is closer to the camera while the other is farther away. Fig. 10.7 shows the variances computed for each LED projection onto the image plane averaged over 3 experimental trials. Consistent with the model (Eqn. 10.6) when $L_s = L_c = 0$, indeed the relationship between the depth (represented by the indices of LED) and the variances is linear with a high correlation coefficient of 0.987. A similar experiment that estimates the depth of a curvy line on a sphere is also shown in Fig. 10.8.

Identifying depth discontinuity. In this experiment, we place two checker-board patterns vertically at two distinct depths (L^{near} and L^{far}) and measure variances of the key points on the scene. We conducted four experiments with different settings of L^{near} and L^{far} (Table 10.1). All were captured in the same setting of $f/11$ with exposure $1/2000$, while the zoom was varied to include the entire scene within the field of view. Fig. 10.9 illustrates the variance discontinuity by two separate parametric fittings of the key points on two checker-boards in one experiment. Clearly, the depth discontinuity can be detected from the variance discontinuity.

Validation of the physics model (Eqn. 10.6). As shown in Fig. 10.9, due to height variations of the turbulence, the variance changes smoothly over the y-axis. However, the variance ratio computed by two points on two checker-boards at the same scan-line is independent of height h, amount of turbulence C_n^2 and aperture diameter D. On the other hand, we can compute the theoretical variance ratios using the ground truth value of L, L_c, L_t and model Eqn. 10.6,. The measurement is consistent with the theory, validating the model in all four settings (Table 10.1)

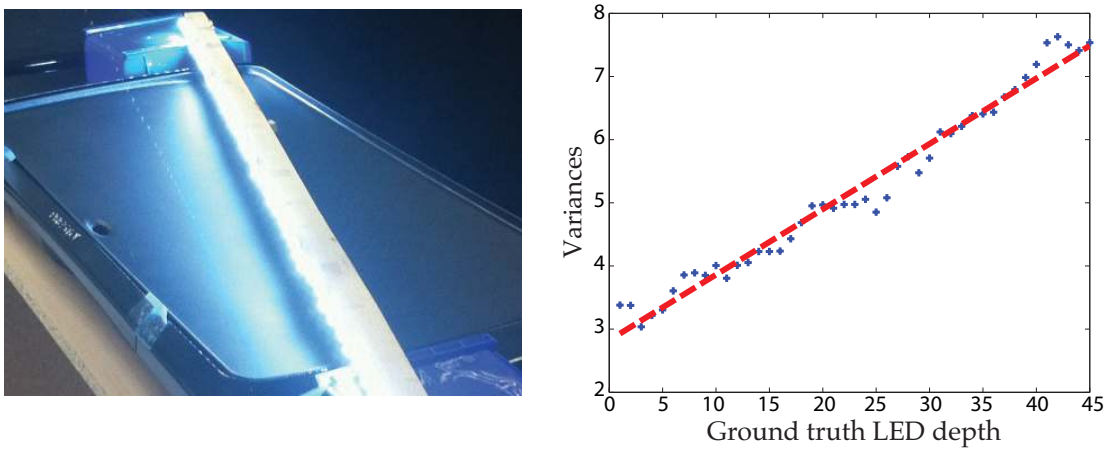
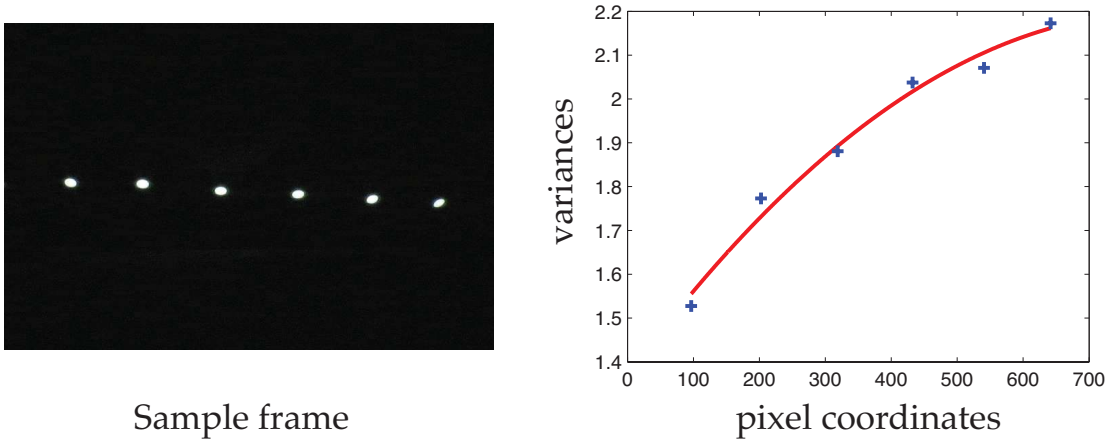


Figure 10.7: **Left:** LEDs are immersed in the turbulence region. **Right:** The relationship between the variance of LED projections and their ground truth depths is very close to linear (correlation coefficient is 0.987), and is consistent with the model (Eqn. 10.6).



Sample frame

Figure 10.8: Ellipse fitting on a video sequence capturing a curve on the sphere through turbulence. Ideally the projection of the LEDs forms an ellipse on the image plane. **Left:** A sample frame of the captured video sequence. **Right:** Average error in fitting is 12.6%. The average fitting error between a covariant x and dependent variable y is computed using $\sqrt{\sum_i (\hat{y}_i - y_i)^2} / \sqrt{\sum_i (\bar{y} - y_i)^2}$, where $\hat{y}_i$ is the fitted value of point x_i and $\bar{y}$ is the mean of y.

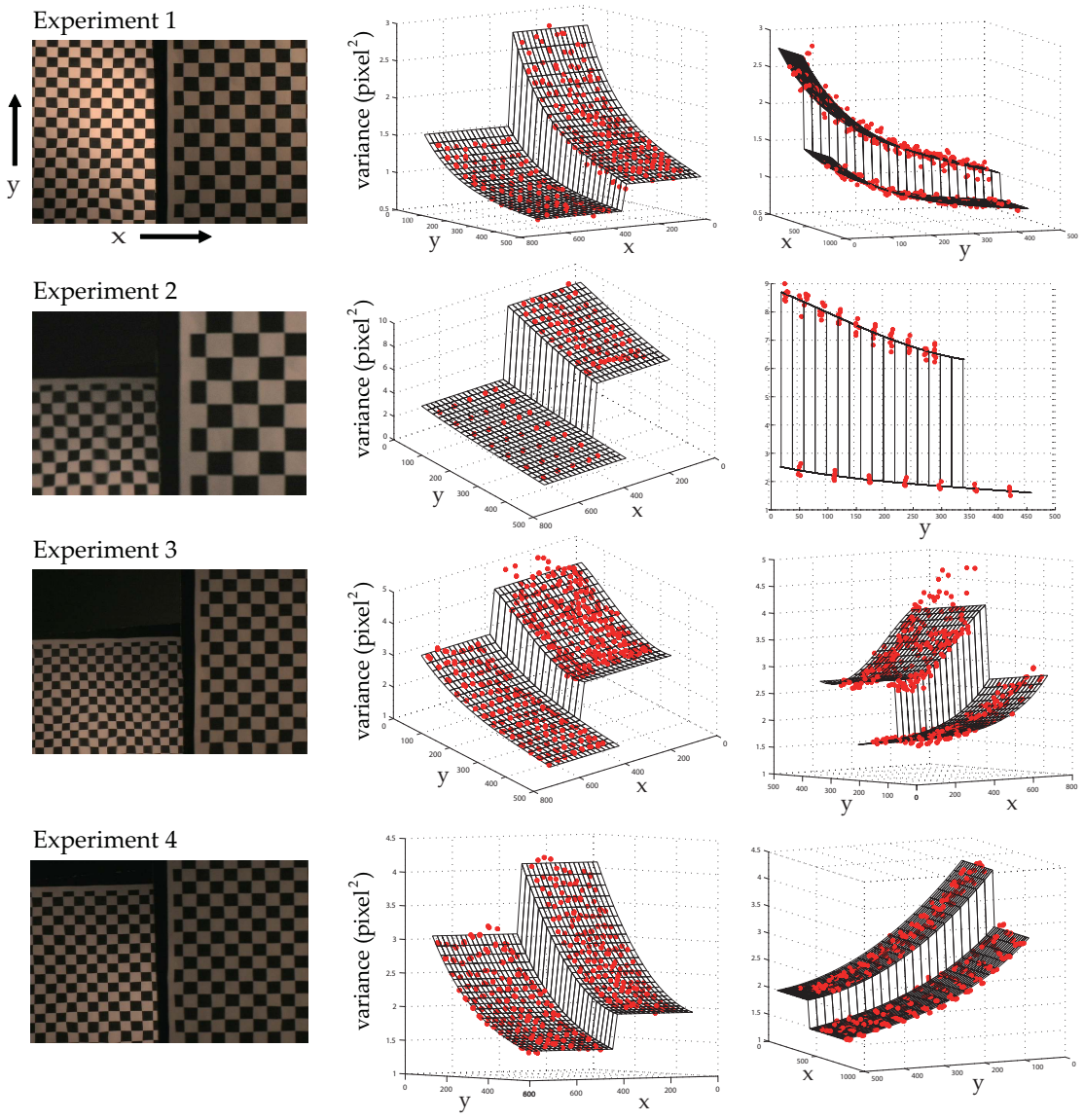


Figure 10.9: Experiments with two planar checker-boards placed at different distances from the camera. The experimental setting can be found in Table 10.1. The first column shows a sample distorted frame, the second and third columns show two views of the variance distribution of the corners of the checker-boards. From the figures, variances changes due to depth discontinuity and height is obvious. We detect the discontinuity and fit smooth surfaces to the variances. The ratio of variances of the two depth planes are then computed and quantitatively compared to the ground truth (Table 10.1).

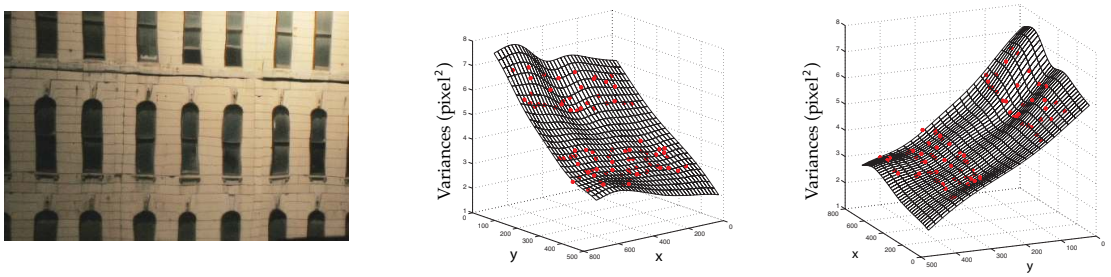


Figure 10.10: Depth estimation of equally spaced points on an inclined planar scene. A sample distorted frame due to turbulence is shown on the left. On the right, are two views of the variance distribution of a sparse set of key points and a smooth function fit illustrating the near planar geometry. From this, it is possible to predict the relative depths of the scene points (assuming the length of the turbulence region to be 1). For evaluation, the relative depth estimated is converted to actual depth by using the actual length (196cm) of turbulence region. The estimated slope of the plane is 0.517 cm per 10 pixels in horizontal direction, compared to the ground truth value 0.529 cm per 10 pixels.

No.	L_c	L_t	L^{near}	L^{far}	Measured	Predicted
Exp1	54	171	163	225	1.77	1.79
Exp2	74	173	183	382	3.60	3.52
Exp3	74	173	247	382	1.67	1.94
Exp4	74	173	247	320	1.56	1.55

Table 10.1: Columns 1-4 show the ground truth measurement (in centimeters) for the four checker-board experiments. Columns 5-6 show the comparison between the measured (5^{th} column) variance ratio and that predicted by the model (6^{th} column). In all but one case, the measurements are very accurate.

that covers both cases where the scene is within and outside the turbulence region.

Depth estimation of equally spaced scene points. We estimate the depths of equally spaced points using the method in Section 10.3. A horizontally slanted plane with a texture of a building facade is placed behind the turbulence region. Fig. 10.10 shows the two views of the computed variances at key points and the surface fits that demonstrate the near planar geometry. Assuming $L_t = 1$ (length of turbulence region), relative depths of scene points can be computed. For validation, the estimated relative depths are converted to absolute ones using the actual length ($L_t = 196$ cm) of the turbulence region. The depth slope of the plane (ΔL in Section 10.3) is estimated as 0.517 cm per 10 pixels in the horizontal direction, compared to the ground truth value of 0.529 cm per 10 pixels.

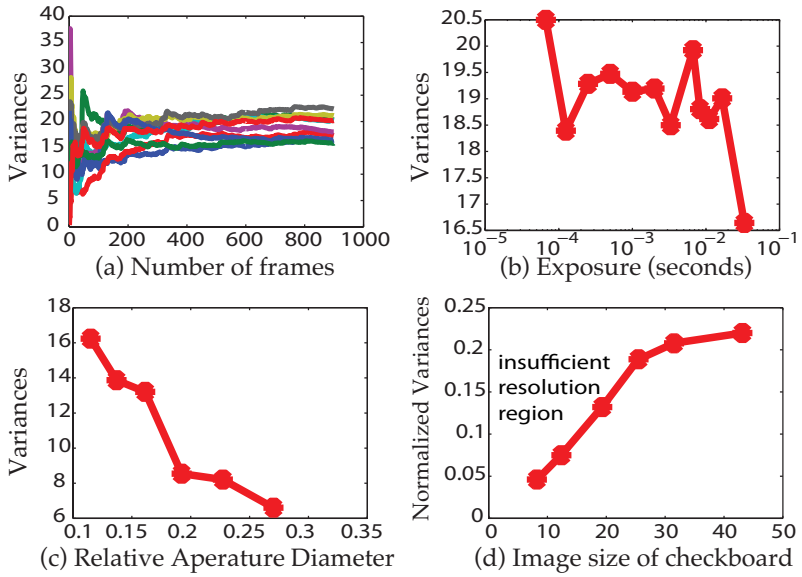


Figure 10.11: Influence of imaging parameters on the variance of the corners of the checker-board pattern. **(a)** The variance estimates converge with sufficient frames, showing that the turbulence is stationary during measurement. **(b)** The measured variance is similar for different exposures, except for very long exposures where the tracking performance degrades due to motion blur. **(c)** Consistent with the model, the measured variance decreases significantly with aperture size. **(d)** Insufficient image resolution results in much lower variance estimation.

10.4.2 Influence of Imaging Parameters

The accuracy of the measured variance depends on many imaging parameters. Larger aperture reduces the depth-of-field, higher exposure time adds unwanted motion blur, and low image magnification causes greater quantization of the variance. Here we present an empirical study.

The estimate of the variance converges as the number of captured frames increases. Fig. 10.11(a) shows the variance of the 10 key points in Exp.1 with different numbers of frames used. In our experiments, stable estimates are achieved using frames captured over 30 seconds using a 30 fps camera. To study the effects of aperture, we vary the $f/\#$ from $f/3.7$ to $f/11$, fix the exposure time at $1/8000s$, and zoom at the highest level. Fig. 10.11(c) shows a significant decrease in variance as predicted by the model in Eqn. 10.6. The plot in Fig. 10.11(b) shows the variances computed by changing only the exposure time. Fig. 10.11(d) shows the effect of varying focal-length. Here we normalize the variances by the pixel size of the checker-board patterns to remove the effects of image magnification. In these experiments, we have tried to maintain the same noise level in the camera by maintaining similar image brightness (by varying illumination intensity). From these plots, aperture size is the main factor that affects the estimation quality. However, since we take variance ratio as a depth measure, the effect of aperture size is reduced

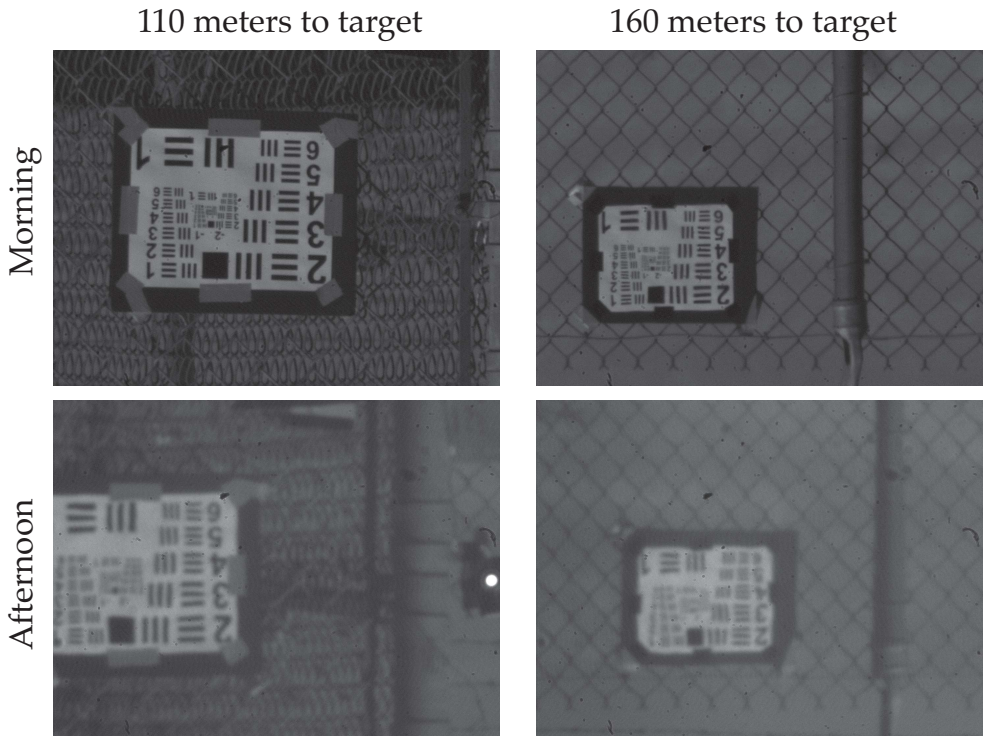


Figure 10.12: Sample frames of the outdoor experiments in the morning and afternoon. The targets are placed at different distances from the camera.

(in theory, independent). Also, for very low magnification (spatial resolution) or long exposure time ($1/30$s), the variance estimate shows large degradation.

In addition, camera shaking can cause false displacements leading to poor variance estimates. In general, this is a hard problem and we will set it aside for future work.

10.5 Outdoor Experiments

Besides indoor experiments in a controlled environment, we also conducted experiments outdoors in a desert region. The imaging setup used for the experiments consists of a Prosilica GC1380H camera and a Celestron C6 Telescope. The focal length is 1500mm. A one-to-one ratio optical relay is used between the telescope and the camera, without changing the focal length of the main telescope. We placed two standard contrast targets 110 meters and 160 meters away from the camera and captured sequences of 300-400 images during mild turbulence (morning) and strong turbulence (afternoon). We used a 30mm aperture in the morning and a 10mm aperture in the afternoon, and varied the exposure times between 0.5ms and 1ms.

We tracked a sparse set of points through each image sequence and rejected outliers such

Morning Capture. Target 110meters away, 30mm aperture.		
0.50ms exposure	0.75ms exposure	1.00ms exposure
4.32 ± 0.49	4.89 ± 0.46	4.79 ± 0.45
Average: 4.67		
Morning Capture. Target 160meters away, 30mm aperture.		
0.50ms exposure	0.75ms exposure	1.00ms exposure
8.23 ± 0.88	7.61 ± 0.67	7.75 ± 0.64
Average: 7.86		

Afternoon Capture. 10mm aperture and 5ms exposure	
Target 110meters away	Target 160meters away
43.68 ± 7.50	69.37 ± 9.49

Table 10.2: Average variance (and its standard derivation) of trackers for outdoor experiment. The variance ratios between 160m and 110m turbulence video are 7.86/4.67=1.6857 (30mm aperture captured in the morning) and 69.37/43.68=1.5785 (10mm aperture captured in the afternoon), close to the distance ratio (160m/110m=1.4545), verifying our model.

as the static trackers of the dirt on the CCD and high-variance erroneous trackers near locally repetitive textures. The computed variances of the trackers converge quickly.

Table 10.2 shows the mean variance computed from all the trackers of each image sequence, for each depth and imaging setting. Since the amount of variance is relatively invariant to exposure change, we further averaged the variance over different exposures. If we take the variance ratio between 110m and 160m, we obtain 1.6857 for 30mm aperture in the morning and 1.5785 for 10mm aperture in the afternoon. Both are close to the ratio of two distances $160/110 = 1.4545$, verifying the dependence of turbulence model on the depth. Besides, Table 10.2 also shows the standard derivation of variances in each video sequence. The low standard deviation shows that the turbulence is indeed homogeneous on surfaces that are perpendicular to the optical axis. Note we do not consider the height variation of turbulence, since compared to the laboratory setting, the target occupies a much narrower field of view.

10.6 Jitter-stereo in Nonhomogeneous Turbulence

Until now, we have addressed depth estimation under homogeneous and simple height-varying turbulence. However, due to unpredictable temperature and humidity fluctuation, turbulence cannot be guaranteed to be homogeneous over a large area and a long period of time. In this case, it is impossible to estimate depth using the model without knowing the turbulence structure function. Instead, we will exploit a general statistical property of turbulence along with a stereo

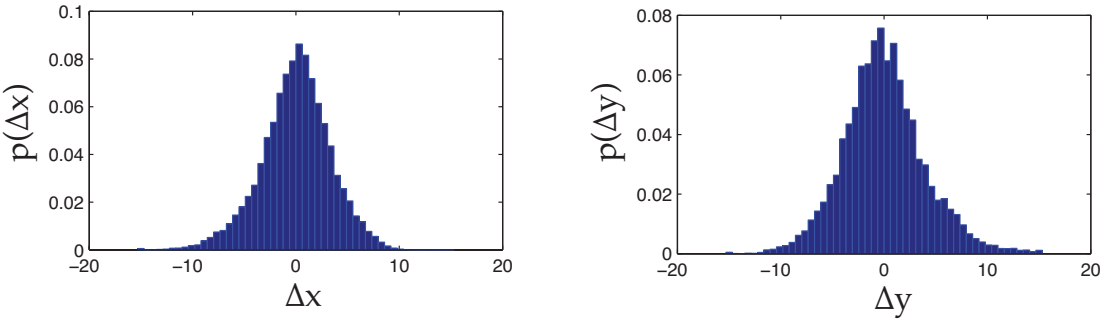


Figure 10.13: Distribution of x and y displacement of trackers in outdoor image sequence, computed over all trackers and all frames from an image sequence in the afternoon. Consistent with our assumption, they follows a zero-mean distribution.

camera pair to improve long-range depth estimation.

Recall that binocular stereo estimates the depth of a scene point by computing scene disparity across two views. If a scene point is far away compared to the stereo baseline, the disparity may be less than one pixel and the depth estimation may fail. However, in the presence of turbulence, the location of the scene point "jitters" around its true location in the image. But how do we estimate the true location of a scene point without turbulence? It has been observed that distribution of (even non-homogeneous) turbulence distortions is close to zero-mean. This is true even if the variance of each scene point is different. Thus, the mean positions of the tracked scene points are their most-likely positions when there is no turbulence. Furthermore, estimating mean locations of trackers allows us to obtain the disparity possibly with sub-pixel accuracy, helping in long-range depth estimation with a short baseline. This approach is also similar in spirit to Swirski et. al [92] that estimates correspondences in stereo using underwater caustics patterns.

In order to experimentally verify our approach, we captured two image sequences of a planar scene 110m away from different view points, with baseline of less than 1m. The two sequences were captured at different times when the turbulence was significantly different. We tracked a common sparse set of scene points in the two views and compute the mean locations. We also verified that the distribution of tracker displacements are zero-mean in all our experiments (see Fig. 10.13). The disparities between the mean-locations of corresponding trackers are then estimated. Note that the disparity of a scene point on a plane is a linear function with respect to its x and y coordinates on the image. The linear fit is strong with a correlation coefficient of 0.976 and is significantly better than computing disparities on a per-frame basis, as shown in Fig. 10.14.

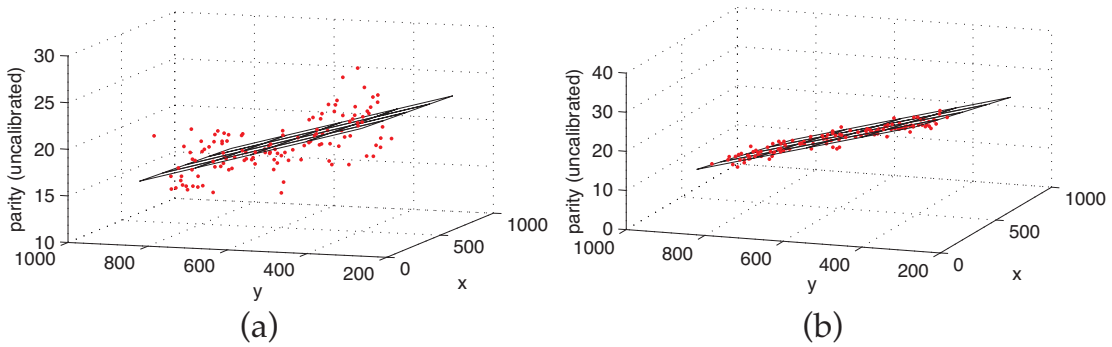


Figure 10.14: Jitter stereo in turbulence. **(a)** (Uncalibrated) disparity computed from two video frames at a certain time, due to turbulence, the disparity is noisy. Over the 319 frames, the correlation coefficient varies from 0.632 to 0.954 with mean being 0.884 and standard derivation being 0.054. **(b)** (Uncalibrated) disparity using mean tracker locations. The disparity is clearly linear (correlation coefficient is 0.976).

10.7 Comparisons with Depth-from-X

This work introduces turbulence as a new cue for scene depth. So, it is instructive to discuss the parallels and differences between depth from turbulence and other depth-from-X approaches in computer vision.

Structure from motion (SFM): SFM relies on estimating the pixel disparity across different views of the scene that reduces with depth. In the case of turbulence, variance of projected scene point is measured from the same camera position over time and monotonically increases with scene distance. Thus, while SFM is suited for shorter distances (for a fixed baseline), depth from turbulence is better in general for a longer range and/or stronger turbulence. But if the scene is outside the turbulence region, the depth precision degrades in the asymptotic region of the variance curve (Fig. 10.5). At the same time, both approaches share the same issues with finding and tracking corresponding features.

Depth from defocus or diffusion (DFD): In both cases, the point-spread function varies across the scene. The extent of the observed blur monotonically increases with distance from the sensor in the case of turbulence and distance from the focal/diffuser plane in the case of DFD [37, 120]. Depth from turbulence requires capture of a temporal sequence of images, and is similar to moving a pinhole across the aperture of the lens to compute depth [1].

Structure from bad weather: Perhaps depth from fog or haze [61] is most similar in spirit to depth from turbulence. These approaches also use a single viewpoint, provide measures that are more reliable for scenes with long distances, and are (mostly) independent of the scene reflectance properties. That said, there are also fundamental differences. Turbulence is a statistical

and temporally varying phenomenon, where depth cues are due to phase variations of the incident light rather than the intensity variations as in fog. The environmental illumination (air light) provides a strong cue for depth from fog, whereas the specific illumination geometry of the environment plays little or no part in depth from turbulence.

Chapter 11

Human Pose Estimation

Human pose estimation is a challenging task in computer vision with many practical applications. For a scenario using a single image, the goal is to estimate the location of each human part. To avoid the curse of dimensionality, one popular approach is to build individual detectors for each human part. Spatial reasoning between parts is then utilized to filter the often noisy responses of the individual detectors. It is critical to design the spatial model so that it captures a versatile yet plausible set of poses.

In this chapter, we use a simplified version of our bottom-up hierarchical framework for this problem. Fig. 11.1 shows our tree-based hierarchy. As mentioned in Sec. 7.2.2, each vertex j contains a position variable $\mathbf{u}_j$ and a type variable z_j. We train the model within a traditional max-margin framework and evaluate it on standard benchmarking dataset. Furthermore, we visually explore the obtained model in two ways: the quality of pose samples and the reconstruction error of recovered poses from labeled testing datasets. First, we sample the pose mixture (type) from the learned model and reconstruct the poses to judge their realism. Most of the sampled poses can be recognized as human-like, while the samples from pairwise tree models [113] are often not natural, showing our model captures global and high-order relationships of poses rather than only local information. Besides, our method reconstructs the pose well and is better than reconstruction using nearest neighbor. This shows that our model can capture a large variation of human poses and generalize well.

Following the seminar David Marr's work [56], many recent works [91, 107] also use hierarchical models for pose estimation. In particular, [91] proposes part types that can be shared among different configurations. In their settings, each mixture component of a latent node also corresponds to one HoG template, modeling the image appearance covered by the descendants of that node. While their main focus is detection, we argue that this is not the ideal strategy for pose estimation, since appearance could vary exponentially with respect to the number of parts,

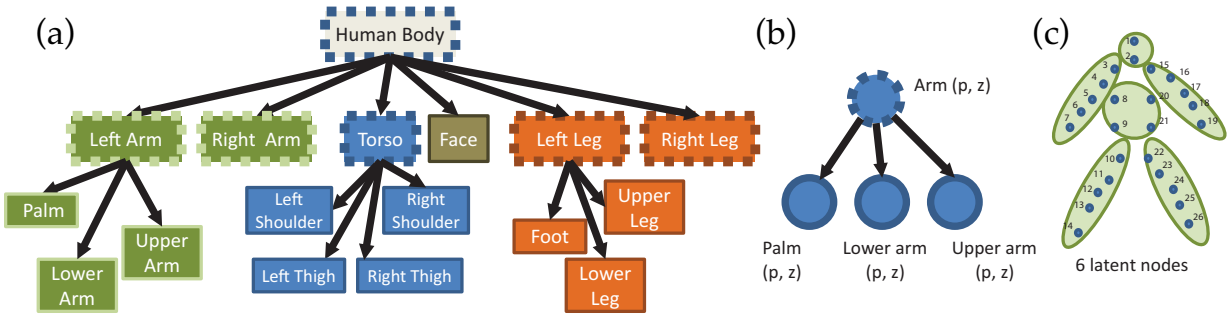


Figure 11.1: **(a)** The hierarchical model for human pose estimation. All nodes with dashed boundaries are latent variables that are on top of body parts (leaves). **(b)** The latent tree model. For any part j, we want to estimate its location $\mathbf{p}_j = (x_j, y_j)$ in the image, and its type z_j. Each type of the object specifies a certain configuration and appearance of that object, as shown in Fig. 11.2(a). **(c)** The 6 latent and 26 leaf nodes in our three-layered hierarchy.

especially for a latent top node (e.g. root node) and cannot be captured by a few mixtures. In our model, only the leaf nodes receive image evidence. The latent nodes handle only geometric deformation and compatibility between parent/child types. Therefore, the number of poses that our model can capture is not the number of mixtures of the root node, but instead the product of mixtures of all nodes.

In terms of numerical evaluation, the performance of our model is on par with the state-of-the-art on three benchmark datasets (PARSE [64], Leeds Sports Dataset [39] and UIUC people [101]). Besides, we also show substantial improvement ($\sim 45\%$) over recent work [107] that builds a hierarchical loopy model for pose estimation.

11.1 Setting up The Hierarchy

We use a hierarchical tree to represent the articulation of human pose, as shown in Fig. 11.1. In this hierarchy of 33 vertices, there are two sets of nodes, the *leaf nodes* (26 vertices) and the *latent nodes* (7 vertices, with dashed outlines). Each leaf node is the primitive body part (i.e., lower arm, upper arm, palm) that has been manually labeled. Each latent node covers a subset of primitive parts that are spatially nearby (i.e. left arm is a latent node that covers lower and upper arms and palm). Finally, the root node represents the entire human body.

As mentioned in Sec. 7.2.2, each vertex V_j has a 2D location variable $\mathbf{u}_j$ and a type variable z_j. This is different from many previous works [3, 22] and similar to [113].

For a *leaf node* such as the palm, its type variable z_i means the appearance of that node could be different, i.e., open/close palm, vertical/horizontal arm (Fig. 11.2(a)). We use type variable

rather than a transformation of a simple template (e.g. rotation/scaling as in [3]), since these transformations usually cannot capture complicated appearance changes, and it is not always necessary to enumerate all the rotation/scalings that are rare in both training and test sets.

For a *latent node* or intermediate node like arm, its type variable z_j means that both the *spatial configuration* and the *preferred types* of its children could be different for different hidden state of the arm (Fig. 11.2(b)). The compatibility between parent type z_j and child type z_k, as usual, can be used to encode constraints such as "an upright arm cannot contain a horizontal lower arm", or "the hand of an upright arm could be open or closed, facing the camera or not", which will be mentioned below.

11.2 Training and Inference

11.2.1 Objective Function

The tree-based hierarchical model defines a graphical model with an objective associated with it:

$$\min_{\{h_i\}} J(\{h_i\}) = \min_{\{h_i\}} \sum_j \sum_{k \in ch(j)} \theta_{jk}(h_j, h_k) + \sum_{k \in [T]} \theta_k(h_k) \tag{11.1}$$

where the pairwise score $\theta_{jk}(h_j, h_k)$ can be written as the summation of the deformation term and the comparability term:

$$\theta_{jk}(h_j, h_k) = \theta^d_{jk}(\mathbf{u}_{j;k}(h_j), \mathbf{u}_k, z_k) + \theta^c_{jk}(z_j, z_k) \tag{11.2}$$

where $\mathbf{u}_{j;k}(h_j) = \mathbf{u}_j + \delta\mathbf{u}_{jk}(z_j)$ is the predicted location from parent j to child k. It consists of two terms θ^d_{jk} and θ^c_{jk}.

The deformation term. The deformation term θ^d_{jk} is defined as:

$$\theta^d_{jk}(\mathbf{u}_{j;k}, \mathbf{u}_k, z_k) = a_k(z_k)\mathrm{dist}(\mathbf{u}_{j;k}, \mathbf{u}_k) \tag{11.3}$$

It computes the misalignment between the predicted location $\mathbf{u}_{j;k}$ and the actual location $\mathbf{u}_k$ of V_k. In the training process, $a_k(z_k)$ is determined from training data. The larger $a_k(z_k)$, the more penalty it will impose on the error between $\mathbf{u}_{j;k}$ and $\mathbf{u}_k$, which corresponds to smaller bound of $g_{jk} = \mathbf{u}_k - \mathbf{u}_{j;k}$.

The compatibility term. The compatibility term $\theta^c_{jk}(z_j, z_k)$ encodes how much the parent type z_j and the child type z_k are compatible, which is a soft version of $z_j \sim z_k$ as defined in Eqn. 7.3 and Eqn. 7.4.

Using types, our model enables appearance to be shared among different latent node (parent) types. For example, templates of open/close hand can be shared in both upright and side-way straight arms. Furthermore, our model can specify what kind of sharing is allowed and what is prohibited (e.g. upright arm cannot have a horizontal lower arm). Such information is encoded in the compatibility term $\theta^c_{jk}(z_j, z_k)$, which is a function between parent type z_j and child type z_k.

On the other hand, appearance sharing in [113] is more restricted. In their model, an open palm pointing up in an upright arm cannot be shared (set as the same type) with an open palm in a side-way straight arm (Fig. 11.2(b), type I vs. type IV). This is because by Eqn. 4 in [113], the predicted relative location of the lower arm is determined by the type of palm. If these two open palms are shared (assigned the same type), then the relative location of the lower arm is forced to be the same, which is not the case in general.

For example (Fig. 11.2(b)), for the constraint "an upright arm cannot contain a horizontal lower arm", we can simply set $\theta^c_{jk}(z_j = \text{upright}, z_k = \text{horizontal lower arm}) = -\infty$. If we want to allow "the hand of an upright arm could be open or closed, facing the camera or not", then we can set $\theta^c_{jk}(z_j = \text{upright}, z_k = \text{open hand}) = \theta^c_{jk}(z_j = \text{upright}, z_k = \text{close hand})$, meaning they are equally probable.

With the compatibility term $\theta^c_{jk}(z_j, z_k)$, one can also model high-order relationships between multiple children. Indeed, by having a common latent parent with large number of mixtures, it is possible to model any joint distribution of children's types and locations.

The Unary term. Besides, for each leaf V_k, there is also an unary term $\theta_k(V_k) = w_k(z_k)^T \phi(I, \mathbf{p}_k)$, where $\phi(I, \mathbf{p}_k)$ is the HoG feature extracted from location $\mathbf{p}_k$ of image I. Note that w_k is a function of z_k, meaning that there is a different template for different type of the part.

11.2.2 Training Procedure

The tree structure of our three-layered hierarchy is shown in Fig. 11.1(c). Given training images with groundtruth location $\mathbf{p}^i(k)$ for leaf node k, for each latent part j (e.g. arm), we first pick one of its child's location as its anchor point (e.g. shoulder), from which we get the location $\mathbf{u}^i_j$. Then we concatenate the relative spatial configuration $\mathbf{u}^i_k - \mathbf{u}^i_j$ of all its children in a vector, cluster them into groups, and estimate the parent-child offsets $\delta\mathbf{u}_{jk}(\cdot)$ accordingly within each group. The clusters are shown in Fig. 11.3. Similarly, we can also build a four-layered hierarchy by subdividing the set of leaf nodes into 2 further subsets.

Given the parent-child relative shifts, we follow the standard max-margin paradigm and use latent SVM to discriminatively and jointly train the hierarchical structure, for both the part de-

tector and the weights for hierarchical model. The formulation is as follows:

$$\begin{aligned} \min \quad & \tfrac{1}{2}\mathbf{w}^T\mathbf{w} + C\textstyle\sum_i \xi_i && (11.4)\\ \text{s.t.} \quad & \max_{V^i} s_i \mathbf{w}^T \Phi(I^i, V^i) \geq 1 - \xi_i && (11.5)\\ & \xi_i \geq 0, a_k(\cdot) \geq 0 && (11.6) \end{aligned}$$

where, $\mathbf{w}$ is the overall weight obtained by concatenating the parameters $w_k(\cdot)$, $a_k(\cdot)$ and $\theta^c_{jk}(\cdot,\cdot)$ $a_k(\cdot)$ together and s_i is the positive/negative label of each image i. Note the weights $a_k(\cdot)$ for the distance have to be non-negative. For all our experiments, we set the regularization constant $C = 0.02$. For optimization, we have used the primal-dual procedure in [113].

11.2.3 Inference Procedure

For efficient inference within the model, all nodes follow a tree structure and standard message passing approach can be used. For node V_j, the incoming message $m_{k\to j}(\mathbf{u}_j, z_j)$ and the outgoing message $m_j(\mathbf{u}_j, z_j)$ are computed as:

$$\begin{aligned} m_j(\mathbf{u}_j, z_j) &= \sum_{k\in ch(j)} m_{k\to j}(\mathbf{u}_j, z_j) && (11.7)\\ m_{k\to j}(\mathbf{u}_j, z_j) &= \min_{z_k}\left[\min_{\mathbf{u}_k} m_k(\mathbf{u}_k, z_k) + \theta^d_{jk}(\mathbf{u}_j, z_j, \mathbf{u}_k)\right] + \theta^c_{jk}(z_j, z_k) && (11.8) \end{aligned}$$

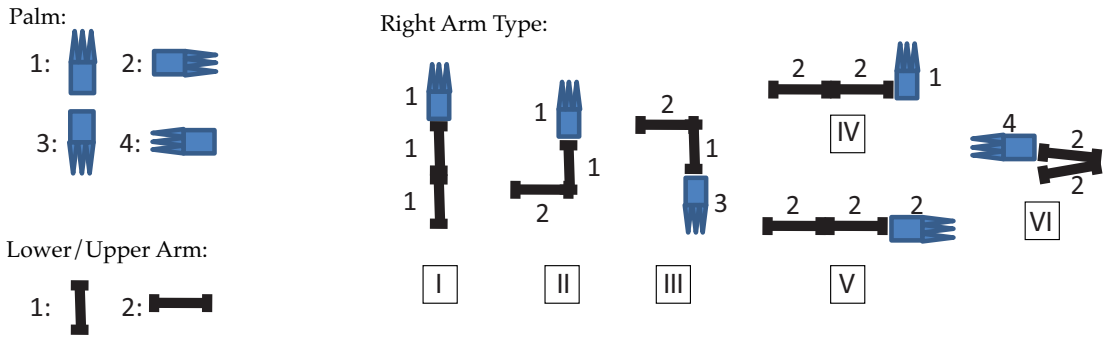
Note that if the parameter $a_k(z_k)$ is only dependent on z_k, then during inference, the message $m^{dt}_k(\mathbf{v}, z_k)$ after distance transform

$$m^{dt}_k(\mathbf{v}, z_k) = \min_{\mathbf{u}_k} m_k(\mathbf{u}_k, z_k) + a_k(z_k)\text{dist}(\mathbf{u}_k, \mathbf{v}) \quad (11.9)$$

can be shared among different z_j and the detection procedure is much faster.

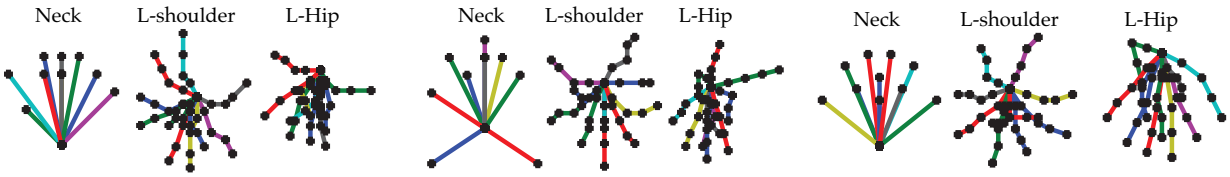
11.3 Empirical Evaluations

Datasets: We use three benchmark datasets to evaluate our approach: PARSE dataset [64], Leeds Sports Dataset (LSP) [39] and UIUC people [101]. PARSE dataset contains 305 images with 100 for trainings and 205 for testing. Leeds Sports Dataset contains 1000 training images and 1000 test images, showing a variety of pose changes. UIUC people dataset contains 346 for training and 247 for testing. Similar to previous works, we use the criterion proposed in [23] for



(a) Leaf nodes (b) Latent nodes

Figure 11.2: **(a)** The model of hand and arm segments and their associated types. **(b)** Six latent configurations of an arm. Note that for different configuration of arms, the appearance model of upper/lower arms and palms can be shared.



(a) PARSE (b) Leeds Sports (c) UIUC people

Figure 11.3: Mixture of each hidden node in the hierarchical model learnt from different datasets. The part labeled on top corresponds to the common node of the mixtures.

performance evaluation, i.e., a part is regarded as correct if its both end-points occur within 50% of the labeled segment length from their true locations.

11.3.1 Exploring the Hierarchical Model

We first study how well our hierarchical model, once trained on a dataset, can represent the spatial configuration of the human body. For this, we (1) sample spatial configurations from the model and observe whether the samples are human-like, and (2) project a given spatial configuration onto the solution space of the model, and check whether the projection (reconstruction) is close to the original configuration.

These two operations show complementary effects of a spatial model. A weakly constrained spatial model, due to its flexibility, may perform extremely well on the reconstruction task, but once sampled, may generate poses that are not human-like. On the other hand, a model with strong spatial prior (e.g. traditional pictorial structure [22]) will generate human-like poses with small variations. But it may fail in reconstructing rare poses accurately. As we shall show, our model achieves a balance between these two criteria.

Sampling. To sample the model, we omit the unary potentials (image evidence), the deformation score $\theta^d_{jk}(\tilde{\mathbf{p}}_k, \mathbf{p}_k, z_k)$ and only use the binary potentials $\theta^c_{jk}(z_j, z_k)$ between parents and children. The truncated score function now becomes:

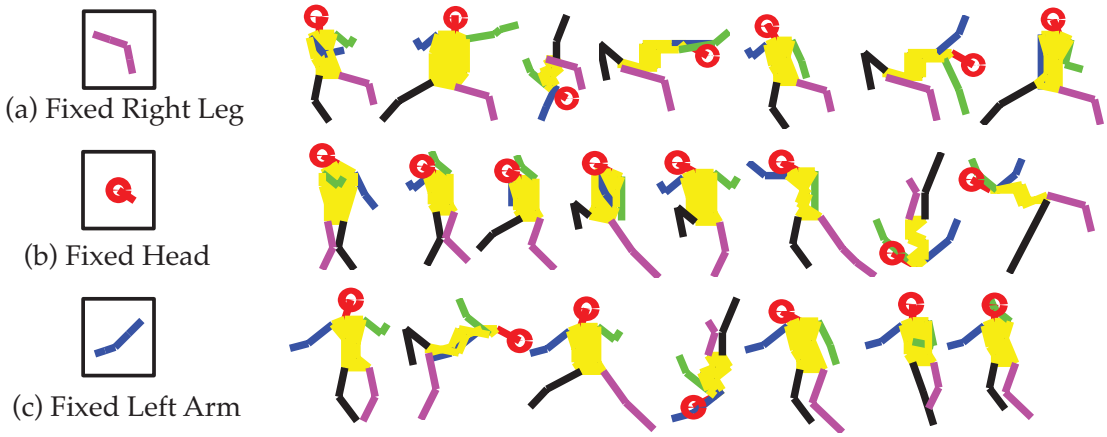
$$J_{\text{trunc}}(\mathbf{Z}) = \sum_j \sum_{k \in ch(j)} \theta^c_{jk}(z_j, z_k) \tag{11.10}$$

where $\mathbf{Z}$ is the collection of all type variables of all nodes. Once samples of $\mathbf{Z}$ are obtained, Eqn. 7.3 (with $g_j = 0$) is used to deterministically generate the human poses in a top-down manner. As a hierarchical tree model, exact sampling is possible.

Fig. 11.6 shows that our model can generate reasonable human-like poses with large variations. In comparison, sampling from [113] often results in weird-looking poses. This means that [113] only encodes local connectivity between nearby parts and does not encode their high-order spatial relationships. In particular, fixing the root type and setting $T \to 0$ gives the most likely pose of that type, as shown in Fig. 11.4. This roughly corresponds to the pose clusters the hierarchical model can capture. However, within each cluster, significant variation is still allowed by changing the type variables below the root node. In contrast, previous works on hierarchical structure [91, 107] all associate a type variable with a template restricting the possible pose variation each node can handle. Alternatively, we can also fix one type of a latent node (e.g. leg, head, arm) and sample the remaining nodes. As shown in Fig. 11.5, our method gives

(a) PARSE (b) Leeds Sports (c) UIUC people

Figure 11.4: The most probable poses of our model for given root types. We can see it encodes a variety of pose changes. In particular, Leeds Sports contain more varied poses than PARSE or UIUC sports.



(a) Fixed Right Leg
(b) Fixed Head
(c) Fixed Left Arm

Figure 11.5: Pose sampling from the hierarchical model learned from Leeds Sports Dataset with one latent node fixed. We can see our model is able to sample human-like poses.

human-like extrapolation of poses.

Reconstruction: For reconstruction, we take one pose from the test sample, allowing the part detectors to fire only at the groundtruth location, and run the detection procedure. Once the best detection is obtained, we can reconstruct the pose with just the type variables but no deformation. Such a reconstructed pose has zero deformation score. The closer the reconstructed pose is to the groundtruth pose, the better the spatial model can fit the given pose.

Our model, once trained on PARSE, can reconstruct poses in the test set of PARSE, better than Nearest-Neighbor. The average error for three-layered structure is 7.19, for four-layered it reduces to 6.55. Our error is lower than 14.63, which is computed from the Nearest-neighbor approach that fits the best global pose in training to the test image. A more flexible model [113] achieves slightly lower error (5.39) for 10 mixture components of each part. For their original model with 5-6 mixtures in each part, the reconstruction error is 7.47. This shows that our model achieves a good balance between creating human-like poses and reconstructability.

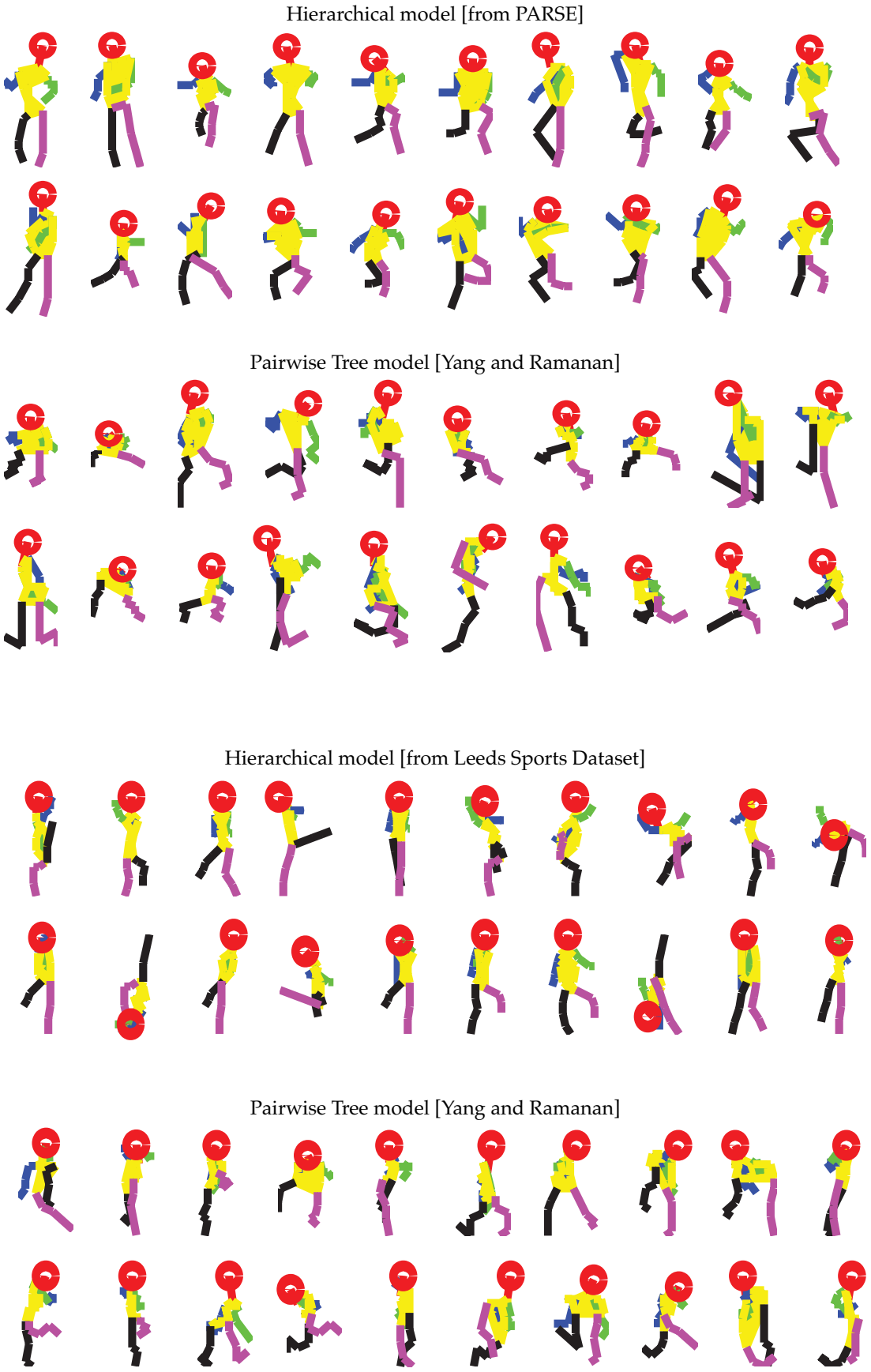


Figure 11.6: Comparison between samples drawn from our hierarchical model and from [113], both learned from PARSE dataset (top) and from Leeds Sports dataset (bottom). Our model captures large variation of poses (especially for Leeds Sports Dataset) and generates reasonable human-like poses.

Dataset	Method	Torso	Head	U. Leg	L. Leg	U. Arm	L. Arm	Total
PARSE	JE [39]	85.4	76.1	73.4	65.4	64.7	46.9	66.2
	YR [113]	**97.6**	**93.2**	83.9	75.1	72.0	**48.3**	**74.9**
	Ours, 3-layer	97.1	92.2	85.1	76.1	71.0	45.1	74.4
	Ours 4-layer	96.1	92.7	81.2	71.0	69.5	39.0	71.0
Leeds	JE [39]	78.1	62.9	65.8	58.8	47.4	32.9	55.1
	Ours (first 200 training)	93.7	86.5	68.0	57.8	49.0	29.2	58.8
	Ours (1000 training, 5 models)	**95.8**	**87.8**	69.9	60.0	51.9	32.9	61.3
	JE [40] (11000 training)	88.1	74.6	**74.5**	**66.5**	**53.7**	**37.5**	**62.7**
UIUC	Wang et.al [107]	86.6	68.8	56.3	50.2	30.8	20.3	47.0
	Our method	**98.8**	**96.8**	**78.7**	**64.2**	**62.2**	**39.5**	**68.5**

Table 11.1: Performance in PARSE, Leeds Sports and UIUC people dataset. For each dataset, the performance of our method is on par with the state-of-art.

11.3.2 Performance on Benchmark Datasets

For PARSE dataset, our performance (in terms of PCP, the percentage of parts being correctly detected) on the test set (205 images) is 73.8%, while [113] achieves 74.9% as reported in their paper. Fig. 11.7 shows that our hierarchical model can handle large pose variations and also often tackle double-counting problem using high-order relationship among parts. See Table 11.1 for the correctly detected ratio for each part.

For Leeds Sports dataset, with only first 200 training images, we achieve 57.9% while the state-of-art achieves 55.2% [39]. Table 11.1 shows the per-part performance.

For UIUC dataset, our hierarchical model outperforms hierarchical poselets [107] by 45%, as shown in Table 11.1. The loopy structure and poselet over large image region may be the reason that hurts their performance. See Table 11.1 for the correctly detected ratio for each part.

As shown in Table 11.1, a deeper hierarchy may hurt the performance since more parameters need to be trained (estimated), resulting in overfitting. However, a shallow one may fail to generalize well since one latent node will be required to encode a large number of joint configurations of children.

From our experiments, we conclude that our methods are on par with the state of the art reported for two datasets and improve significantly over other hierarchical methods. In the future, we will attempt to learn the hierarchical structure automatically from the training data. In terms of applications, we will apply this approach to tracking of human actions in videos.

Figure 11.7: Comparison between our method (Left) and [113] (Right) on some test images of PARSE dataset. Note that our model can handle large pose variation and possibly double-counting.

Figure 11.8: Comparison between our method (Left) and [113] (Right) on Leeds Sports dataset.

Chapter 12

Conclusion and Future Work

This thesis proposes a unified framework and three algorithms with theoretical guarantees to estimate the parameters W of the following deformation model:

$$I(W(\mathbf{x})) = I_0 \tag{12.1}$$

where the deformed image I and the template I_0 are known. The thesis starts from analyzing Eqn. 12.1 in details and then points out three fundamental limitations for conventional approaches, i.e., optimization-based and learning-based approaches:

- *Local Optimum* A local search of a d-dimensional objective function cannot guarantee global optimality.
- *The Nyquist Limits* To achieve worst-case ϵ error, $O(1/\epsilon^d)$ samples are needed to fit a mapping on a d-dimensional manifold.
- *The Curse of Dimensionality* For a mapping on a d-dimensional manifold, the number of samples needed grows exponentially with respect to d.

The reason for these two fundamental limitations is that, traditional approaches fail to use domain-specific knowledge on the level of designing optimization and learning procedures.

For this, this thesis proposes three algorithms, Data-Driven Descent, Top-down and Bottom-up Hierarchical Model, to solve Eqn. 12.1. All approaches have global optimality guarantees under Lipschitz conditions that regulate the appearance difference ΔI of deformed images and the parameter changes $\Delta \mathbf{p}$.

12.1 Data-Driven Descent

The first proposed algorithm, Data-Driven Descent (Chapter 4), gives a solution to Eqn. 12.1 with $O(C^d \log 1/\epsilon)$ sample complexity. It globally parameterizes $W(\mathbf{x})$ as a linear combination of a set of bases: $W(\mathbf{x};\mathbf{p}) = \mathbf{x} + B(\mathbf{x})\mathbf{p}$, The basic idea is to use the group-like structure of deformation: a deformed image is still deformed under an additional deformation. In particular, a deformed image is less deformed if the estimated deformation is close to the true deformation. Using this structure, two different deformation of a specific template can be related, creating additional constraints for the high-dimensional mapping from deformed image to its parameters. The approach is thus applied to water deformation and cloth deformation to estimate local nonrigid deformations.

In general, Data-Driven Descent can be regarded as a procedure to minimize an objective $f(\mathbf{p}, I)$ with respect to $\mathbf{p}$:

$$\mathbf{p}^*(I) = \arg\min_{\mathbf{p}} f(\mathbf{p}, I) \tag{12.2}$$

Here the optimal solution $\mathbf{p}^*(I)$ is a function of "evidence" I. In the scenario of image deformation, $\mathbf{p}$ is the parameters to optimize while I is the deformed image. For single I, $\mathbf{p}^*(I)$ can be found and memorized by a one-time exhaustive search. However, if I is given on the fly, then how to quickly and accurately obtain $\mathbf{p}^*(I)$ becomes a nontrivial task. Optimization on the fly gives suboptimal solutions while a predictive model $I \mapsto \mathbf{p}^*(I)$ requires enormous training samples.

Data-Driven Descent, for the first time, gives a balanced solution to this problem by using the group-like structure of function $f(\mathbf{p}, I)$. From optimization point of view, it would be interesting to check whether other functions have similar properties and can be efficiently optimized with the help of data.

12.2 Top-Down Hierarchical Model

The second proposed algorithm, Top-Down Hierarchical Model (Chapter 6), improves on Data-Driven Descent and gives a solution to Eqn. 12.1 with $O(C_1^d + C_2 \log 1/\epsilon)$ sample complexity. It *locally* parameterizes $W(\mathbf{x})$ as a linear combination of local bases, each with a finite spatial support. While the global deformation is first estimated using nearest neighbor like Data-Driven Descent, finite supports of patches enable the deformation to be *factorized* into local patches when the residual deformations becomes smaller. As a result, the sample complexity drops double-exponentially when the algorithm proceeds, yielding a reduced sample complexity. Furthermore, the required Lipschitz conditions are also relaxed since large patches only need to give

a rough estimation while only the local patches take care of the final landmark precision.

In general, the Top-Down Hierarchical Model coincides with a *context-dependent* independence in machine learning: a set of variables $\{\mathbf{p}(k)\}$ are independent of each other given evidence I only in a local region $\|\mathbf{p}\|_\infty \leq C_0$, and is highly correlated elsewhere. In Top-Down Hierarchical Model, all $\mathbf{p}(k)$s are deformations at different local patches. Such structures are prevailing in decision trees, in which left and right branches of trees splitting different region of the space, lead to totally different ways of handling data. The difference is that in our model, such structures are discovered in the model and hard-wired into the algorithm, while in decision trees the structures are statistically learned from substantially more training samples.

As a future work, I believe context-dependent independence relationship is abundant in other models and can be formally discovered after detailed analysis. Conceptually, it is like "divide-and-conquer" in algorithm design. This helps us in finding a suitable approach for this problem and may largely reduce their sample complexity.

12.3 Bottom-Up Hierarchical Model

The Bottom-Up Hierarchical Model proposed in Chapter 7 further reduces the sample complexity by building a layerwise invariant representation that gradually reduces the degrees of freedom (DoFs) from bottom to up, by discarding information unrelated to high-level decisions. This improves on Top-Down Model whose top-most layer accumulates all the DoFs and dominates sample complexity, and finally breaks the curse of dimensionality.

Furthermore, in the bottom-up model, Lipschitz conditions are only assumed on the leaf node, while in the top-down case, Lipschitz conditions are assumed true at every patch of different scales and layers. This makes the bottom-up model more appealing since its assumptions are more likely to be true. The reason is that in the bottom-up model, top-level representations are computed from bottom levels and can be fully characterized quantatively by relating the Lipschitz constants $(\alpha_j, \gamma_j, r_j)$ for parent-child pairs.

It is not very satisfactory that Alg. 8 is proposed with nice theoretical guarantees while for practical problems like human pose estimation, a traditional tree-based graphical model is used. As a future work, finding a training procedure to Alg. 8 is critical. The meaning of training here, is to find a suitable hierarchical structure, a set of parameters including $\bar{g}_j$, $\delta\mathbf{u}_{jk}(\cdot)$, anchor point $ah(j)$, weights on aggregation operator w_{jk} (See Eqn. 8.6), so that the Lipschitz conditions (Assumption 7.3.1) and global optimal conditions (Eqn. 7.110) hold.

It seems that we now have a lot to optimize. Fortunately, with the help of our analysis in Sec. 7.3.2, parameters all have clear meanings. For example, according to the analysis, param-

eters such as $\delta\mathbf{u}_{jk}(\cdot)$ and $\bar{g}_j$ are only related to the deformation field itself, and has nothing to do with the template image I_0. On the other hand, the choice of anchor point $ch(j)$ and the aggregation weights w_{jk} are related to the distinctiveness and reliability of local part detections. With these insights, the training procedure will become intuitive and decoupled.

On the contrary, current deep learning research trains billions of parameters jointly using a huge optimization framework, without knowing the meaning of each parameters. This could be potentially harmful due to two factors. First, the optimization may involve a search over hierarchical structure, whose search space is enormously large. Second, even a good model is found by a massive and yet lucky search, no insights into the model can be obtained. It would be humiliating to declare that the human brain has been copied to the computer, and still we humans do not know how it works.

Our bottom-up framework is not restricted to deformation modeling. For image classification or scene understanding, the same intuition holds as well. Local and ambiguous information is gathered in the local region and sent to the top level, where a global decision is made and is propagated downwards. The key here is to determine which components are to be sent and which are to be discarded.

12.4 Application-Specific Deformation modeling

Chapter 9 introduces a new system for correction of document warping using a single image. Our method can be used to reconstruct and rectify documents of arbitrary smooth shapes, not restricted to ruling surfaces as in previous works [51, 117]. In addition, it is language and content independent and can deal with texts written in English, Chinese and Hindi. Many future works follow from this work. Boosting the process speed of the system is crucial in time-sensitive applications such as mobile OCR. For book scanning that involve a video sequence of text images, incorporating temporal information is important for faster processing and more accurate warping estimation. Systematically modeling interactions between the horizontal and vertical directions can potentially lead to more robust estimation. Finally, strong perspective is critical for successful 3D reconstruction. In the case of weak perspective, more information, e.g., shading, can be incorporated to yield better results.

Chapter 10 provides an initial attempt at understanding what depth cues can be extracted from optical turbulence. We derived a simple relation between scene depths and variance of the projected scene points under turbulence. Our experiments showed, somewhat surprisingly, that accurate depth cues can be obtained from optical turbulence. There are several avenues of future work including dense scene reconstruction and image super-resolution from the image

sequence under turbulence. We wish to also study several other related physical phenomena. The twinkling of stars is caused primarily due to the changes in amplitude of the incident wave that are distance-related as well. Aside from temperature gradients, the chaotic movement of a medium itself can cause turbulence. This type of phenomenon can occur due to under water currents, due to strong wind flow in the upper atmosphere, and due to air flow around engines. We wish to build upon this work to apply to these scenarios.

Part IV

Appendix

Chapter 13

Appendix A: Sampling within a Hypercube

Many of the theorems in this thesis are based on a design of sampling strategy so that for every location $\mathbf{p}$ in the hypercube $[-r, r]^D$, there exists at least one sample sufficiently close to it. Furthermore, we want to minimize the number of samples needed for this design. Mathematically, we want to find the smallest *cover* of $[-r, r]^D$.

In the following, we provide one necessary and two sufficient conditions. The first is for the general case (covering $[-r, r]^D$ entirely), while the second specifies the number of samples needed if $\mathbf{p}$ is known to be on a low-dimensional subspace, in which we could have better bounds.

13.1 Covering the Entire Hypercube

Theorem 13.1.1 (Sampling Theorem, Necessary Conditions) *To cover $[-r, r]^D$ with smaller hypercubes of side $2\alpha r$ ($\alpha < 1$), at least $\lfloor 1/\alpha^D \rfloor$ hypercubes are needed.*

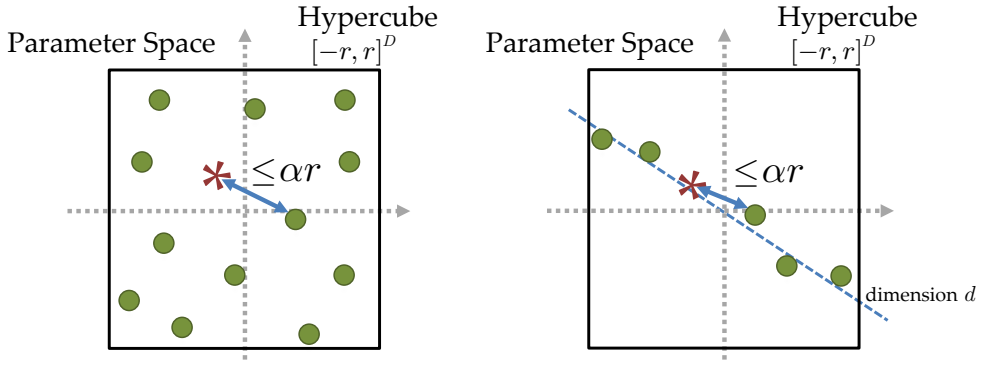
Proof The volume of $[-r, r]^D$ is $\mathrm{Vol}(r) = (2r)^D$, while the volume of each hypercube of side $2\alpha r$ is $\mathrm{Vol}(2\alpha r) = (2r)^D \alpha^D$. A necessary condition of covering is the total volume of small hypercube has to be at least larger than $\mathrm{Vol}(r)$:

$$N\mathrm{Vol}(2\alpha r) \geq \mathrm{Vol}(r) \tag{13.1}$$

which gives:

$$N \geq \frac{\mathrm{Vol}(r)}{\mathrm{Vol}(2\alpha r)} = \frac{1}{\alpha^D} \geq \left\lfloor \frac{1}{\alpha^D} \right\rfloor \tag{13.2}$$

∎



(a) Uniform Sampling (b) Sample on deformation subspace

Figure 13.1: Sampling strategies for Thm. 13.1.2 and Thm. 13.2.2. **(a)** Uniform sampling within a hypercube $[-r,r]^D$ so that for any $\mathbf{p} \in [-r,r]^D$, there exists at least one training sample that is αr close to $\mathbf{p}$. **(b)** If we know that in addition to the constraint $\|\mathbf{p}\|_\infty \leq r$, $\mathbf{p}$ always lies on a subspace of dimension $d < D$, then just assigning samples near the subspace within the hypercube suffices.

Theorem 13.1.2 (Sampling Theorem, Sufficient Conditions) *With $\lceil 1/\alpha \rceil^D$ number of samples ($\alpha < 1$), for any $\mathbf{p}$ contained in the hypercube $[-r,r]^D$, there exists at least one sample $\hat{\mathbf{p}}$ so that $\|\hat{\mathbf{p}} - \mathbf{p}\|_\infty \leq \alpha r$.*

Proof Uniformly distribute the training samples within the hypercube does the job. In particular, denote

$$n = \left\lceil \frac{1}{\alpha} \right\rceil \tag{13.3}$$

Thus we have $1/n = 1/\lceil 1/\alpha \rceil \leq 1/(1/\alpha) = \alpha$. We put training sample of index $(i_1, i_2, \ldots, i_d)$ on d-dimensional coordinates:

$$\hat{\mathbf{p}}_{i_1,i_2,\ldots,i_d} = r\left[-1 + \frac{2i_1 - 1}{n}, -1 + \frac{2i_2 - 1}{n}, \ldots, -1 + \frac{2i_D - 1}{n}\right] \tag{13.4}$$

does the job. Here $1 \leq i_k \leq n$ for $k = 1 \ldots D$. So each dimension we have n training samples. Along the dimension, the first sample is r/n distance away from $-r$, then the second sample is $2r/n$ distance to the first sample, until the last sample that is r/n distance away from the boundary r. Then for any $\mathbf{p} \in [-r,r]^D$, there exists i_k so that

$$\left| \mathbf{p}(k) - r\left(-1 + \frac{2i_k - 1}{n}\right) \right| \leq \frac{1}{n} r \leq \alpha r \tag{13.5}$$

This holds for $1 \leq k \leq D$. As a result, we have

$$\|\mathbf{p} - \hat{\mathbf{p}}_{i_1,i_2,\ldots,i_D}\|_\infty \leq \alpha r \tag{13.6}$$

and the total number of samples needed is $n^D = \lceil 1/\alpha \rceil^D$.

13.2 Covering a Subspace within Hypercube

Now we consider the case that $\mathbf{p}$ lies on a subspace of dimension d, i.e., there exists a column-independent matrix U of size D-by-d so that $\mathbf{p} = U\mathbf{h}$ for some hidden variable $\mathbf{h}$. This happens if we use overcomplete local bases to represent the deformation. Since each landmark is related to two local bases, usually $D/2$ number of landmarks will give the deformation parameters $\mathbf{p}$ with apparent dimension D.

In this case, we do not need to fill the entire hypercube $[-r, r]^D$. In fact, we expect the number of samples to be exponential with respect to only d rather than D.

Definition 13.2.1 (Noise Controlled Deformation Field) *A deformation field $\mathbf{p}$ is called* noise-controlled deformation of order k and expanding factor c, *if for every $\mathbf{p} \in [-r, r]^D$, there exists a k-dimensional vector ($k \geq d$) $\mathbf{v} \in [-r, -r]^k$ so that $\mathbf{p} = f(\mathbf{v})$. Furthermore, for any $\mathbf{v}_1, \mathbf{v}_2 \in [-r, r]^k$, we have:*

$$\|\mathbf{p}_1 - \mathbf{p}_2\|_\infty = \|f(\mathbf{v}_1) - f(\mathbf{v}_2)\|_\infty \leq c\|\mathbf{v}_1 - \mathbf{v}_2\|_\infty \tag{13.7}$$

for a constant $c \geq 1$.

Note that by the definition of intrinsic dimensionality d, $\mathbf{v}$ could be only d-dimensional and still $\mathbf{p} = f(\mathbf{v})$. However, in this case, c could be pretty large. In order to make c smaller, we can have a *redundant* k-dimensional representation $\mathbf{h}$ with $k > d$.

Many global deformation field satisfies Definition 13.2.1. For example, an affine deformation field $\mathbf{p}$ defined on a grid has $d = 6$ and $k = 8$, no matter how many landmarks ($D/2$) there are. This is because each component of $\mathbf{p}$ can be written as

$$\mathbf{p}(k) = [\lambda_1 x_k + \lambda_2 y_k + \lambda_3, \lambda_4 x_k + \lambda_5 y_k + \lambda_6] \tag{13.8}$$

for location $\mathbf{l}_k = (x_k, y_k)$. Therefore, since any landmarks $\mathbf{l}_k$ within a rectangle can be linearly represented by the locations of four corners in a convex manner, the deformation vector $\mathbf{p}(k)$ on $\mathbf{l}_k$ can also be linearly represented by the deformation vectors of four corners (8 DoF):

$$\mathbf{p}(k) = A_k \mathbf{v} = \sum_{j=1}^{4} a_{kj} \mathbf{v}(j) \tag{13.9}$$

with $\mathbf{v}$ is the concatenation of four deformation vectors from the four corners, $0 \leq a_{kj} \leq 1$ and $\sum_j a_{kj} = 1$. For any $\mathbf{p} \in [-r, r]^D$, $\mathbf{v}$ can be found by just picking the deformation of its four

corners, and thus $\|\mathbf{v}\|_\infty \leq r$. Furthermore, we have for $\mathbf{v}_1, \mathbf{v}_2 \in [-r, r]^k$:

$$\|\mathbf{p}_1 - \mathbf{p}_2\|_\infty = \|f(\mathbf{v}_1) - f(\mathbf{v}_2)\|_\infty \leq \max_k \sum_{j=1}^{4} a_{kj} \|\mathbf{v}_1(j) - \mathbf{v}_2(j)\| \leq \|\mathbf{v}_1 - \mathbf{v}_2\|_\infty \tag{13.10}$$

Therefore, $c = 1$.

Similarly, for deformation that contains pure translation and rotation ($d = 3$), we just pick displacement vectors on two points ($k = 4$), the rotation center and the corner as $\mathbf{v}$. Then we have:

$$\begin{aligned} \mathbf{p}(r, \theta) &= \mathbf{p}_{\text{center}} + \frac{r}{r_{\text{corner}}} R(\theta)(\mathbf{p}_{\text{corner}} - \mathbf{p}_{\text{center}}) && (13.11) \\ &= (I - \frac{r}{r_{\text{corner}}} R(\theta))\mathbf{p}_{\text{center}} + \frac{r}{r_{\text{corner}}} R(\theta)\mathbf{p}_{\text{corner}} && (13.12) \end{aligned}$$

where I is the identity matrix, $R(\theta)$ is the 2D rotational matrix and r_{corner} is the distance from the center location to the corner. Here we reparameterize the landmarks with polar coordinates (r, θ). Therefore, for two different $\mathbf{v}_1$ and $\mathbf{v}_2$, since $r \leq r_{\text{corner}}$, we have:

$$\begin{aligned} \|\mathbf{p}_1(r, \theta) - \mathbf{p}_2(r, \theta)\|_\infty &\leq \left\| (I - \frac{r}{r_{\text{corner}}} R(\theta))(\mathbf{p}_{\text{center},1} - \mathbf{p}_{\text{center},2}) \right\|_\infty && (13.13) \\ &+ \left\| \frac{r}{r_{\text{corner}}} R(\theta)(\mathbf{p}_{\text{corner},1} - \mathbf{p}_{\text{corner},2}) \right\|_\infty && (13.14) \\ &\leq 2\|\mathbf{p}_{\text{center},1} - \mathbf{p}_{\text{center},2})\|_\infty + \sqrt{2}\|\mathbf{p}_{\text{corner},1} - \mathbf{p}_{\text{corner},2})\|_\infty && (13.15) \\ &\leq (2 + \sqrt{2})\|\mathbf{v}_1 - \mathbf{v}_2\|_\infty && (13.16) \end{aligned}$$

since $|\cos(\theta)| + |\sin(\theta)| \leq \sqrt{2}$. Therefore,

$$\|\mathbf{p}_1 - \mathbf{p}_2\|_\infty = \max_{r,\theta} \|\mathbf{p}_1(r, \theta) - \mathbf{p}_2(r, \theta)\|_\infty \leq (2 + \sqrt{2})\|\mathbf{v}_1 - \mathbf{v}_2\|_\infty \tag{13.17}$$

So $c = 2 + \sqrt{2} \leq 3.5$.

Given this definition, we thus have the following sampling theorem for deformation parameters $\mathbf{p}$ lying on a subspace that is noise-controlled.

Theorem 13.2.2 (Sampling Theorem, Sufficient Condition for Subspace Case) *For any noise-controlled deformation field $\mathbf{p} = f(\mathbf{v})$ with order k and expanding factor c, with $c_{SS}\lceil 1/\alpha \rceil^d$ number of training samples distributed in the hypercube $[-r, r]^D$, there exists at least one sample $\hat{\mathbf{p}}$ so that $\|\hat{\mathbf{p}} - \mathbf{p}\|_\infty \leq \alpha r$. Note $c_{SS} = \lceil c \rceil^k \left\lceil \frac{1}{\alpha} \right\rceil^{k-d}$.*

Proof We first apply Thm. 13.1.2 to the hypercube $[-r, r]^k$. Then with $\lceil \frac{c}{\alpha} \rceil^k$ samples, for any

$\mathbf{v} \in [-r, r]^k$, there exists a training sample $\mathbf{v}^i$ so that

$$\|\mathbf{v} - \mathbf{v}^i\|_\infty \leq \frac{\alpha r}{c} \tag{13.18}$$

We then build the training samples $\{\mathbf{p}^i\}$ by setting $\mathbf{p}^i = f(\mathbf{v}^i)$. Therefore, from the definition of noise cancelling, given any $\mathbf{p} \in [-r, r]^D$, there exists an $\mathbf{v} \in [-r, r]^k$ so that $\mathbf{p} = f(\mathbf{v})$. By the sampling procedure, there exists $\mathbf{v}^i$ so that $\|\mathbf{v} - \mathbf{v}^i\|_\infty \leq \frac{\alpha}{c} r$, and therefore:

$$\|\mathbf{p} - \mathbf{p}^i\|_\infty \leq c\|\mathbf{v} - \mathbf{v}^i\|_\infty \leq \alpha r \tag{13.19}$$

setting $\hat{\mathbf{p}} = \mathbf{p}^i$ thus does the job. Finally, note that

$$\left\lceil \frac{c}{\alpha} \right\rceil^k \leq \lceil c \rceil^k \left\lceil \frac{1}{\alpha} \right\rceil^{k-d} \left\lceil \frac{1}{\alpha} \right\rceil^d \tag{13.20}$$

So setting $c_{SS} = \lceil c \rceil^k \left\lceil \frac{1}{\alpha} \right\rceil^{k-d}$ suffices (since $\lceil ab \rceil \leq \lceil a \rceil \lceil b \rceil$). ∎

Bibliography

[1] E.H. Adelson and J.Y.A. Wang. Single lens stereo with a plenoptic camera. *IEEE Trans. on PAMI*, 14(2), 1992.

[2] A. Agarwal and B. Triggs. Recovering 3D human pose from monocular images. *PAMI*, 28(1):44–58, 2006.

[3] M. Andriluka, S. Roth, and B. Schiele. Pictorial structures revisited: People detection and articulated pose estimation. In *ter Vision and Pattern Recognition*, pages 1014–1021. IEEE, 2009.

[4] Bradley Atcheson, Ivo Ihrke, Wolfgang Heidrich, Art Tevs, Derek Bradley, Marcus Magnor, and Hans-Peter Seidel. Time-resolved 3d capture of non-stationary gas flows. *ACM TOG*, 27(5), 2008.

[5] S. Baker and I. Matthews. Equivalence and efficiency of image alignment algorithms. In *CVPR*, 2001.

[6] S. Baker and I. Matthews. Lucas-kanade 20 years on: A unifying framework. *IJCV*, 56 (3):221–255, 2004.

[7] An Approach Based, Maurizio Pilu, and Maurizio Pilu. Deskewing perspectively distorted documents: An approach based on perceptual organization. *HP White Paper*, 2001.

[8] M. Bergtholdt, J. Kappes, S. Schmidt, and C. Schnörr. A study of parts-based object class detection using complete graphs. *International journal of computer vision*, 87(1):93–117, 2010.

[9] A. Bissacco, M.H. Yang, and S. Soatto. Fast human pose estimation using appearance and motion via multi-dimensional boosting regression. In *CVPR*, 2007.

[10] Fred L. Bookstein. Principal warps: Thin-plate splines and the decomposition of deformations. *PAMI*, 1989.

[11] M.S. Brown and W.B. Seales. Document restoration using 3D shape: a general deskewing algorithm for arbitrarily warped documents. In *ICCV*. IEEE Computer Society, 2001.

[12] M.S. Brown and W.B. Seales. Image restoration of arbitrarily warped documents. *PAMI*, 2004.

[13] H. Cao, X. Ding, and C. Liu. Rectifying the bound document image captured by the camera: A model based approach. *ICDAR*, 2003.

[14] X. Cao, Y. Wei, F. Wen, and J. Sun. Face alignment by explicit shape regression. In *CVPR*, 2012.

[15] Xudong Cao, Yichen Wei, Fang Wen, and Jian Sun. Face Alignment by Explicit Shape Regression. In *CVPR*, 2012.

[16] T.F. Cootes, G.J. Edwards, and C.J. Taylor. Active appearance models. In *ECCV*, 1998.

[17] A. Donate and E. Ribeiro. Improved Reconstruction of Images Distorted by Water Waves. In *International Conference on Computer Vision Theory and Applications*, pages 228–235. Springer, 2006.

[18] A. Donate, G. Dahme, and E. Ribeiro. Classification of Textures Distorted by Water-Waves. In *Proceedings of the 18th International Conference on Pattern Recognition-Volume 02*, pages 421–424. IEEE Computer Society Washington, DC, USA, 2006.

[19] A. Ecker, K.N. Kutulakos, and A.D. Jepson. Shape from planar curves: A linear escape from flatland. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE, 2007.

[20] Alexei A. Efros, Volkan Isler, Jianbo Shi, and Mirko Visontai. Seeing through water. In *Neural Information Processing Systems (NIPS 17)*, 2004.

[21] H. Ezaki, S. Uchida, A. Asano, and H. Sakoe. Dewarping of document image by global optimization. In *ICDAR*, pages 302–306, 2006.

[22] P.F. Felzenszwalb and D.P. Huttenlocher. Efficient matching of pictorial structures. In *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, volume 2, pages 66–73. IEEE, 2000.

[23] V. Ferrari, M. Marin-Jimenez, and A. Zisserman. Progressive search space reduction for human pose estimation. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.

[24] M.A. Fischler and R.A. Elschlager. The representation and matching of pictorial structures. *Computers, IEEE Transactions on*, 100(1):67–92, 1973.

[25] David Forsyth. Shape from texture and integrability. In *ICCV*, 2001.

[26] RT Framkot and R. Chellappa. A method for enforcing integrability in shape from shading

algorithms. *PAMI*, pages 439–451, 1988.

[27] DL Fried. Probability of getting a lucky short-exposure image through turbulence. *Optical Society of America, Journal*, 68:1651–1658, 1978.

[28] Jérôme Gilles, Tristan Dagobert, and Carlo Franchis. Atmospheric turbulence restoration by diffeomorphic image registration and blind deconvolution. In *ACIVS*, 2008.

[29] M. Gleicher. Projective registration with difference decomposition. In *CVPR*, 1997.

[30] GD Hager and PN Belhumeur. Efficient region tracking with parametric models of geometry and illumination. *PAMI*, 20(10):1025–1039, 1998.

[31] S Harmeling, M. Hirsch, S. Sra, and Schlkopf. Online blind deconvolution for astronomy. In *ICCP*, 2009.

[32] J. Hays, M. Leordeanu, A. Efros, and Y. Liu. Discovering texture regularity as a higher-order correspondence problem. *Computer Vision–ECCV 2006*, pages 522–535, 2006.

[33] G.T. Herman. *Image reconstruction from projections*. Academic Press, New York, 1980.

[34] R. Hess and A. Fern. Improved video registration using non-distinctive local image features. In *CVPR*, 2007.

[35] M. Hirsch, S. Sra, B. Schlkopf, and S. Harmeling. Efficient filter flow for space-variant multiframe blind deconvolution. In *CVPR*, 2010.

[36] X.W. Hou, S.Z. Li, H.J. Zhang, and Q.S. Cheng. Direct appearance models. In *CVPR*, 2001.

[37] T. Hwang, J.J. Clark, and A.L. Yuille. A depth recovery algorithm using defocus information. In *CVPR*, 1989.

[38] Akira Ishimaru. *Wave Propagation and Scattering in Random Media*. Wiley-IEEE Press, 1999.

[39] S. Johnson and M. Everingham. Clustered pose and nonlinear appearance models for human pose estimation. In *BMVC*, 2010.

[40] S. Johnson and M. Everingham. Learning effective human pose estimation from inaccurate annotation. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 1465–1472. IEEE, 2011.

[41] N. Joshi and M.F. Cohen. Seeing mt. rainier: Lucky imaging for multi-image denoising, sharpening and haze removal. *In ICCP*, 2010.

[42] O.V. Ju, VV Savchenko, IM Levin, et al. Correction of image distorted by wavy water surface: laboratory experiment. In *Proceedings of the IV International Conference Current*

Problems in Optics of Natural Waters(ON W'2007), N. Novgorod, pages 91–93, 2007.

[43] F. Jurie and M. Dhome. Hyperplane approximation for template matching. *PAMI*, pages 996–1000, 2002.

[44] A.N Kolomogorov. *Dissipation of energy in the locally isotropic turbulence*. Mathematical and Physical Sciences, 1941.

[45] A.N Kolomogorov. *The local structure of turbulence in incompressible viscous fluid for very large Reynold's numbers*. Comptes rendus de l'Acadmie des Sciences de l'U.R.S.S. 32: 16C18, 1941.

[46] H. Koo and N. Cho. State estimation in a document image and its application in text block identification and text line extraction. *ECCV*, 2010.

[47] Norman S. Kopeika. *A System Engineering Approach to Imaging*. SPIE Publications, 1999.

[48] Alex Krizhevsky, Ilya Sutskever, and Geoff Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

[49] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR*, 2006.

[50] J. Liang, D. DeMenthon, and D. Doermann. Unwarping Images of Curved Documents Using Global Shape Optimization. *CBDAR*, 2005.

[51] J. Liang, D. DeMenthon, and D. Doermann. Geometric rectification of camera-captured document images. *PAMI*, 2007.

[52] Y. Liu, R.T. Collins, and Y. Tsin. A computational model for periodic pattern perception based on frieze and wallpaper groups. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(3):354–371, 2004.

[53] D.G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004.

[54] B.D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *IJCAI*, volume 81, pages 674–679, 1981.

[55] J. Malik and R. Rosenholtz. Computing local surface orientation and shape from texture for curved surfaces. *IJCV*, 23(2):149–168, 1997. ISSN 0920-5691.

[56] D. Marr and H.K. Nishihara. Representation and recognition of the spatial organization of three-dimensional shapes. *Proceedings of the Royal Society of London. Series B. Biological Sciences*, 200(1140):269–294, 1978.

[57] I. Matthews and S. Baker. Active appearance models revisited. *IJCV*, 60(2):135–164, 2004.

[58] Markus Moll and Luc Van Gool. Optimal templates for non-rigid surface reconstruction. In *ECCV*, 2012.

[59] N.J.W. Morris. Image-based water surface reconstruction with refractive stereo. Master's thesis, University of Toronto, 2004.

[60] N.J.W. Morris and K.N. Kutulakos. Dynamic refraction stereo. In *ICCV*, 2005.

[61] S.G. Narasimhan and S.K. Nayar. Vision and the atmosphere. *IJCV*, 48(3), 2002.

[62] M.H. Nguyen and F. De la Torre. Local minima free parameterized appearance models. In *CVPR*, 2008.

[63] M. Park, K. Brocklehurst, R.T. Collins, and Y. Liu. Deformed lattice detection in real-world images using mean-shift belief propagation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(10):1804–1816, 2009.

[64] D. Ramanan. Learning to parse images of articulated bodies. In *NIPS*, volume 19, page 1129, 2007.

[65] H. Richard, M. Raffel, M. Rein, J. Kompenhans, and G.E.A. Meier. Demonstration of the applicability of a background oriented schlieren method. In *Intl. Symp. on Applications of Laser Techniques to Fluid Mechanics*, 2000.

[66] G. Rogez, J. Rihan, S. Ramalingam, C. Orrite, and P.H.S. Torr. Randomized trees for human pose detection. In *CVPR*, 2008.

[67] Michael C. Roggemann and Byron Welsh. *Imaging Through Turbulence*. CRC press, 1 edition, March 1996.

[68] R. Rosales and S. Sclaroff. Learning body pose via specialized maps. In *NIPS*, 2002.

[69] D. Rueckert, LI Sonoda, C. Hayes, D.L.G. Hill, M.O. Leach, and D.J. Hawkes. Nonrigid registration using free-form deformations: application to breast MR images. *Medical Imaging*, 18(8):712–721, 1999.

[70] M. Salzmann and P. Fua. Reconstructing sharply folding surfaces: A convex formulation. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 1054–1061. IEEE, 2009.

[71] M. Salzmann and R. Urtasun. Combining discriminative and generative methods for 3d deformable surface and articulated pose reconstruction. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 647–654. IEEE, 2010.

[72] M. Salzmann, R. Hartley, and P. Fua. Convex optimization for deformable surface 3-d tracking. In *ICCV*, 2007.

[73] M. Salzmann, F. Moreno-Noguer, V. Lepetit, and P. Fua. Closed-form solution to non-rigid 3d surface registration. In *European conference on computer vision*, pages 581–594, 2008.

[74] Mathieu Salzmann and Pascal Fua. Reconstructing Sharply Folding Surfaces: A Convex Formulation. In *CVPR*, 2009.

[75] Mathieu Salzmann and Raquel Urtasun. Combining Discriminative and Generative Methods for 3D Deformable Surface and Articulated Pose Reconstruction. In *CVPR*, 2010.

[76] S. Sclaroff and J. Isidoro. Active blobs. In *ICCV*, 1998.

[77] Paul Scovanner, Saad Ali, and Mubarak Shah. A 3-dimensional sift descriptor and its application to action recognition. In *Proceedings of the 15th international conference on Multimedia*, pages 357–360. ACM, 2007.

[78] Thomas Serre, Lior Wolf, and Tomaso Poggio. Object recognition with features inspired by visual cortex. In *CVPR*, 2005.

[79] G. Settles. *Shlieren and Shadowgraph Techniques*. Springer-Verlag, 2001.

[80] A. Shaji, A. Varol, L. Torresani, and P. Fua. Simultaneous point matching and 3d deformable surface reconstruction. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 1221–1228. IEEE, 2010.

[81] R. Shefer, M. Malhi, and A. Shenhar. Waves distortion correction using cross correlation, 2001.

[82] A. Shekhovtsov, I. Kovtun, and V. Hlavác. Efficient MRF deformation model for non-rigid image matching. *Computer Vision and Image Understanding*, 112(1):91–99, 2008. ISSN 1077-3142.

[83] J. Shi and C. Tomasi. Good features to track. In *CVPR*, 1994.

[84] M. Shimizu, S. Yoshimura, M. Tanaka, and M. Okutomi. Super-resolution from image sequence under influence of hot-air optical turbulence. In *CVPR*, 2008.

[85] Aaron P. Shon, Keith Grochow, Aaron Hertzmann, and Rajesh P. N. Rao. Learning Shared Latent Structure for Image Synthesis and Robotic Imitation. In *NIPS*, 2006.

[86] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-time human pose recognition in parts from single depth images. In *CVPR*, volume 2, page 7, 2011.

[87] Jamie Shotton, John Winn, Carsten Rother, and Antonio Criminisi. Textonboost for image understanding: Multi-class object recognition and segmentation by jointly modeling texture, layout, and context. *International Journal of Computer Vision*, 81(1):2–23, 2009.

[88] H.Y. Shum and R. Szeliski. Systems and experiment paper: Construction of panoramic image mosaics with global and local alignment. *IJCV*, 36(2):101–130, 2000.

[89] L. Sigal, A. Balan, and M.J. Black. Combined discriminative and generative articulated pose and non-rigid shape estimation. In *NIPS*, 2007.

[90] C. Sminchisescu, A. Kanaujia, and D.N. Metaxas. BM^{3}E: Discriminative Density Propagation for Visual Tracking. *PAMI*, 2007.

[91] M. Sun and S Savarese. Articulated part-based model for joint object detection and pose estimation. In *ICCV*, 2011.

[92] Y. Swirski, Y.Y. Schechner, B. Herzberg, and S. Negahdaripour. Stereo from flickering caustics. In *ICCV*, 2009.

[93] V.I Tatarskii. *Wave Propagation in a Turbulent Medium*. McGraw-Hill Books, 1961.

[94] J. Taylor, A.D. Jepson, and K.N. Kutulakos. Non-rigid structure from locally-rigid motion. In *CVPR*. IEEE, 2010.

[95] Jonathan Taylor, Allan Jepson, and Kyros Kutulakos. Non-Rigid Structure from Locally-Rigid Motion. In *CVPR*, 2010.

[96] A. Thayananthan, R. Navaratnam, B. Stenger, P.H.S. Torr, and R. Cipolla. Pose estimation and tracking using multivariate regression. *Pattern Recognition Letters*, 29(9):1302–1310, 2008.

[97] T.P. Tian and S. Sclaroff. Fast globally optimal 2d human detection with loopy graph models. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 81–88. IEEE, 2010.

[98] Yuandong Tian and Srinivas G. Narasimhan. Seeing through Water: Image Restoration using Model-based Tracking. In *ICCV*, 2009.

[99] Yuandong Tian and Srinivasa Narasimhan. A globally optimal data-driven approach for image distortion estimation. In *CVPR*, 2010.

[100] Yuandong Tian and Srinivasa G. Narasimhan. Globally optimal estimation of nonrigid image distortion. *IJCV*, 2012.

[101] D. Tran and D. Forsyth. Improved human parsing with a full relational model. *Computer Vision–ECCV 2010*, pages 227–240, 2010.

[102] Oncel Tuzel, Fatih Porikli, and Peter Meer. Learning on Lie Groups for Invariant Detection and Tracking. In *CVPR*, 2008.

[103] A. Ulges, C.H. Lampert, and T. Breuel. Document capture using stereo vision. In *ACM symposium on Document Engineering*, 2004.

[104] Gaurav Vasudeva, Damon R. Honnery, and Julio Soria. Non-intrusive measurement of a density field using the background oriented schlieren method. In *4th Australian Conf. on Laser Diagnostic in Fluid Mechanics and Combustion*, 2005.

[105] L. Venkatakrishnan and G. E. A. Meier. Density measurements using the background oriented schlieren technique. *Experiments in Fluids*, 37:237–247, 2004.

[106] Y. Wang, S. Lucey, and J. Cohn. Enforcing convexity for improved alignment with constrained local models. In *CVPR*, 2008.

[107] Y. Wang, D. Tran, and Z. Liao. Learning hierarchical poselets for human parsing. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 1705–1712. IEEE, 2011.

[108] ZY Wen, D. Fraser, A. Lambert, and HD Li. Reconstruction of Underwater Image by Bispectrum. In *IEEE International Conference on Image Processing, 2007. ICIP 2007*, volume 3, 2007.

[109] G. Wetzstein, W. Heidrich, and R. Raskar. Hand-held schlieren photography with light field probes. In *ICCP*, 2011.

[110] A.P. Witkin. Recovering surface shape and orientation from texture. *Artificial Intelligence*, 1981.

[111] C. Wu and G. Agam. Document image de-warping for text/graphics recognition. *Structural, Syntactic, and Statistical Pattern Recognition*, pages 243–253, 2009.

[112] A. Yamashita, A. Kawarago, T. Kaneko, and K.T. Miura. Shape reconstruction and image restoration for non-flat surfaces of documents with a stereo vision system. In *ICPR*, 2004.

[113] Y. Yang and D. Ramanan. Articulated pose estimation with flexible mixtures-of-parts. In *Computer Vision and Pattern Recognition (CVPR)*, pages 1385–1392. IEEE, 2011.

[114] L. Yuan, J. Sun, L. Quan, and H.Y. Shum. Image deblurring with blurred/noisy image pairs. In *ACM Transactions on Graphics (TOG)*, volume 26, page 1. ACM, 2007.

[115] A. Zandifar. Unwarping scanned image of Japanese/English documents. In *ICIAP*, 2007.

[116] Yun Zeng, Chaohui Wang, Yang Wang, Xianfeng Gu, Dimitris Samaras, and Nikos Paragios. Dense Non-rigid Surface Registration Using High-Order Graph Matching. In *CVPR*,

2010.

[117] L. Zhang and C.L. Tan. Warped image restoration with applications to digital libraries. In *ICDAR*, 2005.

[118] Z. Zhang, C.L. Tan, and L. Fan. Restoration of curved document images through 3d shape modeling. In *CVPR*, 2004.

[119] X. Zhao, H. Ning, Y. Liu, and T. Huang. Discriminative estimation of 3D human pose using gaussian processes. In *ICPR*, 2008.

[120] C. Zhou, O. Cossairt, and S. Nayar. Depth from diffusion. In *CVPR*, 2010.

[121] Y. Zhou, E. Yeniaras, P. Tsiamyrtzis, N. Tsekos, and I. Pavlidis. Collaborative tracking for mri-guided robotic intervention on the beating heart. *MICCAI*, 2010.

[122] X. Zhu and P. Milanfar. Stabilizing and deblurring atmospheric turbulence. *In ICCP*, 2011.