Inférence efficace et apprentissage des modèles graphiques pour la segmentation des formes multi-organes

Haithem Boussaid

ECOLE CENTRALE PARIS

P H D T H E S I S

to obtain the title of

Doctor of Science

of ECOLE CENTRALE PARIS

Specialty : Applied Mathematics

Defended by

Haithem Boussaid

Efficient Inference and Learning in Graphical Models for Multi-organ Shape Segmentation

prepared at Ecole Centrale Paris, Center for Visual Computing

defended on January $8^{th}, 2015$

Jury :

Reviewers :	Hervé Delingette	-	INRIA Sophia-Antipolis
	Adrian Barbu	-	Florida State University
	Jean-Philippe Thiran	-	EPFL Lausanne
Supervisors :	Iasonas Kokkinos	-	Ecole Centrale Paris
	Nikos Paragios	-	Ecole Centrale Paris
President :	Chris Taylor	-	University of Manchester
examiners :	Florence Forbes	-	INRIA Grenoble
	Dhruv Batra	-	Virginia Tech
Invited :	Daniel Duclos	-	SAFRAN Group

Acknowledgements

This thesis is the achievement of efforts of many individuals. May these words acknowledge their investment and contributions.

First, I would like to express my deepest gratitude to my thesis supervisor Prof. *Iasonas Kokkinos*. His passion and devotion to computer vision and his rigor at work in the tiniest detail have been to me an ultimate source of admiration, inspiration and motivation. *Iasonas* has invested a lot of his time to guide me, to discuss ideas and to deeply improve my work, during this 4-year journey while being permanently supporting and friendly. I never forget our meetings in a daily basis at the approach of deadlines, when he showed up as an inspiring workmate more than an advisor. *Iasonas* has been a role model as a supervisor and a scientist; and I learned a lot from him, such us how to find ideas interesting to computer vision problems, how to write scientific papers, prepare slides, present works orally and how to manage time. For this, I am forever grateful to him and I believe that he took up my skills and expertise to the next level.

I am equally thankful to my co-supervisor and mentor Prof. *Nikos Paragios* for his unconditional support and all the opportunities he offered to me since I came to the Center for Visual Computing, including this thesis. I consider myself fortunate to have worked with *Nikos*; his endless generosity, excellent human qualities and investment in helping me at a professional level as well as at a personal level made me consider him as a close friend to me. I would like to thank him for having a critical eye on my work, for his continuous suggestions to improve it and for his piece of advice, to cope with the thesis and in life in general. I owe *Nikos* a debt of gratitude and it is my pride, pleasure and honor to having collaborated with him during these years.

I would like to thank all the members of my thesis committee: the reviewers *Hervé Delingette*, *Jean-Philippe Thiran* and *Adrian Barbu*, the chairman *Chris Taylor* and the examiners *Florence Forbes* and *Dhruv Batra* for taking time to read and evaluate my thesis. It is a privilege to have such well-known researchers consider my work.

I would like to express my gratitude to Prof. *Christos Davatzikos* for hosting me in his group in the university of Pennsylvania during a summer internship. I met there with great people and it was a pleasure to discuss with them on medical image analysis problems and experience American lifestyles. I am particularly thankful to *Aristeidis Sotiras*, *Guray Erus* and *Meng-Kang Hsieh* who facilitated my stay in Philadelphia.

I would like to express my sincere gratitude to all my colleagues in the lab for providing a great work atmosphere and for helping me in various ways. I thank former PhD students who set the bar high with the quality of their work

and with whom I had passionate discussions. They are *Chaohui Wang*, *Loic Simon*, *Ahmed Besbes*, *Olivier Teboul*, *Radhouene Neji*, *Martin de la Gorce*, *Nicolas Honnorat*, *Fabrice Michel* and *Salma Essafi*. I am particularly grateful to *Chaohui Wang* and *Loic Simon* with whom I had the chance to collaborate with. I also thank my current PhD fellows *Stavros Alchatzidis*, *Stavros Tsogkas*, *Eduard Trulls*, *Puneet Kumar*, *Enzo Ferrante*, *Vivian Fecamp*, *Stefan Kinauer*, *Siddhartha Chandra*, *Eugene Belilovsky*, *Maxim Berman* and *Evgenios Kornaropoulos*. I wish you all the best. In particular, I would like to address special thanks to my officemate *Stavros Tsogkas*. Our administrative assistant *Natalia Leclercq* deserves also my gratitude for facilitating my paperwork and her kindness.

On a personal note, I would like to take this opportunity to thank my parents for their support in tough moments despite the distances; I thank my father *Zouhair* for pushing me to follow my dreams no matter what sacrifices it takes. I thank my mother *Jamila* for her permanent cheerfulness and her unconditional love.

I am grateful to my beloved wife *Sonia*, who made the foolish decision to marry a permanently stressed PhD student. I would like to thank her for all her sacrifices, support and patience especially during weekends and late-night hours when I had to work to meet a deadline. She provided me with care, love and joy and laugh moments that made me forget about my research for a while.

I thank also my parents in law *Bouraoui* and *Najet* for their understanding and for keeping willingly my daughter during my internship in the United States, and especially my mother in law *Najet* who came to Paris to keep my daughter when I went to Beijing for a conference.

Many thanks go to my longtime buddies *Ahmed*, *Baha* and *Moez* for being indeed true friends.

Finally, I dedicate this thesis to my daughter *Lyn*, who since her birth last year gave meaning to my life.

Abstract

This thesis explores the use of discriminatively trained deformable contour models (DCMs) for shape-based segmentation in medical images. We make contributions in two fronts: in the learning problem, where the model is trained from a set of annotated images, and in the inference problem, whose aim is to segment an image given a model. We demonstrate the merit of our techniques in a large X-Ray image segmentation benchmark, where we obtain systematic improvements in accuracy and speedups over the current state-of-the-art.

For learning, we formulate training the DCM scoring function as *large-margin structured prediction* and construct a training objective that aims at giving the highest score to the ground-truth contour configuration. We incorporate a loss function adapted to DCM-based structured prediction. In particular, we consider training with the *Mean Contour Distance* (MCD) performance measure. Using this loss function during training amounts to scoring each candidate contour according to its Mean Contour Distance to the ground truth configuration. Training DCMs using structured prediction with the standard zero-one loss already outperforms the current state-of-the-art method [Seghers *et al.* 2007] on the considered medical benchmark [Shiraishi *et al.* 2000, van Ginneken *et al.* 2006]. We demonstrate that training with the MCD structured loss further improves over the generic zero-one loss results by a statistically significant amount.

For inference, we propose efficient solvers adapted to combinatorial problems with discretized spatial variables. Our contributions are three-fold: first, we consider inference for loopy graphical models, making no assumption about the underlying graph topology. We use an efficient decomposition-coordination algorithm to solve the resulting optimization problem: we decompose the model's graph into a set of open, chain-structured graphs. We employ the *Alternating Direction Method of Multipliers* (ADMM) to fix the potential inconsistencies of the individual solutions. Even-though ADMM is an approximate inference scheme, we show empirically that our implementation delivers the exact solution for the considered examples. Second, we *accelerate optimization* of chain-structured graphical models by using the Hierarchical A^* search algorithm of [Felzenszwalb & Mcallester 2007] coupled with the pruning techniques developed in [Kokkinos 2011a]. We achieve a one order of magnitude speedup in average over the state-of-the-art technique based on Dynamic Programming (DP) coupled with Generalized Distance Transforms (GDTs) [Felzenszwalb & Huttenlocher 2004]. Third, we incorporate the Hierarchical A^* algorithm in the ADMM scheme to guarantee an efficient optimization of the underlying chain structured subproblems. The

resulting algorithm is naturally adapted to solve the loss-augmented inference problem in structured prediction learning, and hence is used during training and inference.

In Appendix A, we consider the case of 3D data and we develop an efficient method to find the mode of a 3D kernel density distribution. Our algorithm has guaranteed convergence to the global optimum, and scales logarithmically in the volume size by virtue of recursively subdividing the search space. We use this method to rapidly initialize 3D brain tumor segmentation where we demonstrate substantial acceleration with respect to a standard mean-shift implementation.

In Appendix B, we describe in more details our extension of the Hierarchical A^* search algorithm of [Felzenszwalb & Mcallester 2007] to inference on chain-structured graphs.

Keywords:

Shape Segmentation, Deformable Contour Models, ADMM, Structured Prediction, Hierarchical A^* , Kernel Density Estimation.

Résumé

Cette thèse explore l'utilisation des modèles de contours déformables pour la segmentation basée sur la forme des images médicales. Nous apportons des contributions sur deux fronts: dans le problème de l'apprentissage statistique, où le modèle est formé à partir d'un ensemble d'images annotées, et le problème de l'inférence, dont le but est de segmenter une image étant donnée un modèle. Nous démontrons le mérite de nos techniques sur une grande base d'images à rayons X, où nous obtenons des améliorations systématiques et des accélérations par rapport à la méthode de l'état de l'art.

Concernant l'apprentissage, nous formulons la formation de la fonction de score des modèles de contours déformables en un problème de prédiction structurée à grande marge et construisons une fonction d'apprentissage qui vise à donner le plus haut score à la configuration vérité-terrain. Nous intégrons une fonction de perte adaptée à la prédiction structurée pour les modèles de contours déformables. En particulier, nous considérons l'apprentissage avec la mesure de performance consistant en la distance moyenne entre contours, comme une fonction de perte. L'utilisation de cette fonction de perte au cours de l'apprentissage revient à classer chaque contour candidat selon sa distance moyenne du contour vérité-terrain. Notre apprentissage des modèles de contours déformables en utilisant la prédiction structurée avec la fonction zéro-un de perte surpasse la méthode [Seghers *et al.* 2007] de référence sur la base d'images médicales considérée [Shiraishi *et al.* 2000, van Ginneken *et al.* 2006]. Nous démontrons que l'apprentissage avec la fonction de perte de distance moyenne entre contours améliore encore plus les résultats produits avec l'apprentissage utilisant la fonction zero-un de perte et ce d'une quantité statistiquement significative.

Concernant l'inférence, nous proposons des solveurs efficaces et adaptés aux problèmes combinatoires à variables spatiales discrétisées. Nos contributions sont triples: d'abord, nous considérons le problème d'inférence pour des modèles graphiques qui contiennent des boucles, ne faisant aucune hypothèse sur la topologie du graphe sous-jacent. Nous utilisons un algorithme de décomposition-coordination efficace pour résoudre le problème d'optimisation résultant: nous décomposons le graphe du modèle en un ensemble de sous-graphes en forme de chaines ouvertes. Nous employons la *Méthode de direction alternée des multiplicateurs* (ADMM) pour réparer les incohérences des solutions individuelles. Même si ADMM est une méthode d'inférence approximative, nous montrons empiriquement que notre implémentation fournit une solution exacte pour les exemples considérés. Deuxièmement, nous accélérons l'optimisation des modèles graphiques en forme de chaîne en utilisant l'algorithme de recherche hiérarchique A^* [Felzenszwalb & Mcallester 2007]

couplé avec les techniques d'élagage développés dans [Kokkinos 2011a]. Nous réalisons une accélération de 10 fois en moyenne par rapport à l'état de l'art qui est basé sur la programmation dynamique (DP) couplé avec les transformées de distances généralisées [Felzenszwalb & Huttenlocher 2004]. Troisièmement, nous intégrons A^* dans le schéma d'ADMM pour garantir une optimisation efficace des sous-problèmes en forme de chaine. En outre, l'algorithme résultant est adapté pour résoudre les problèmes d'inférence augmentée par une fonction de perte qui se pose lors de l'apprentissage de prédiction des structure, et est donc utilisé lors de l'apprentissage et de l'inférence.

Dans l'annexe A, nous considérons le cas des images 3D et nous développons une méthode efficace pour trouver le mode d'une distribution à noyau de densité en 3D. Notre algorithme a une convergence garanti vers l'optimum global, et une complexité logarithmique en fonction de la taille du volume grâce à la subdivision récursive l'espace de recherche. Nous utilisons cette méthode pour initialiser rapidement la segmentation 3D de tumeurs cérébrales où nous démontrons une accélération substantielle par rapport à une implémentation standard de l'algorithme mean shift.

Dans l'annexe B, nous décrivons plus en détails notre implémentation de l'algorithme A^* hiérarchique de [Felzenszwalb & Mcallester 2007].

Contents

List of Figures

List of Tables

Chapter 1

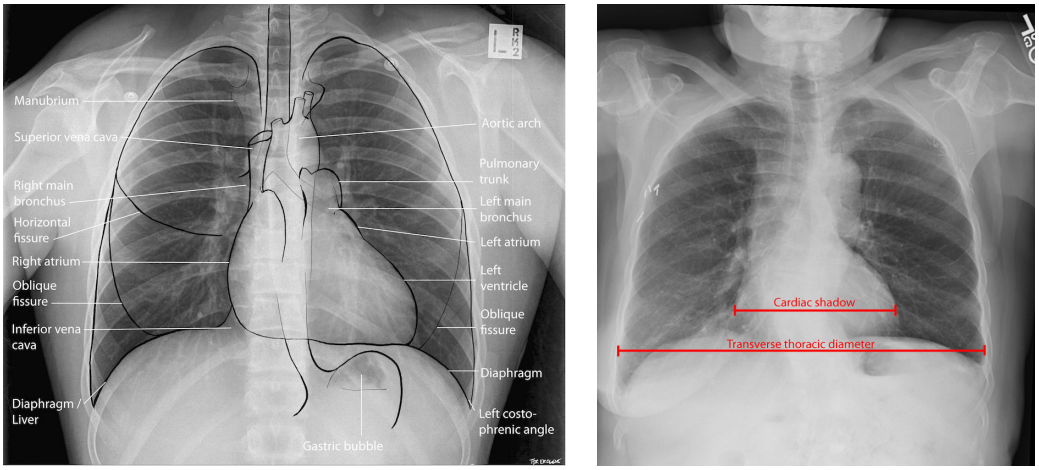
Introduction

Contents

1.1 Context and Motivation

The automatic localization of shapes in medical images is of paramount importance in a host of medical image analysis applications, involving anatomical object segmentation [van Ginneken *et al.* 2002, Seghers *et al.* 2007, Heimann & Meinzer 2009], tracking [Paragios 2003], registration [Ellingsen *et al.* 2010] and atlas building [Durrleman *et al.* 2012]. These computer-aided tasks are valuable for physicians as they help to diagnose diseases and avoid tedious manual tasks. For instance the segmentation of the lung fields and the heart from a chest radiograph enables radiologists to measure the cardiothoratic ratio -shown in Figure 1.1- for cardiomegaly diagnosis [Nakamori *et al.* 1991, Ishida *et al.* 2005]. In this case, the segmentation is cast as finding the outer contour of each object of interest.

Shape extraction in medical images is challenging. Images produced by medical image acquisition systems -such as X-rays, Computed Tomography, Magnetic Resonance Imaging and ultrasounds- often suffer from low contrast, missing boundaries and non-discriminative object appearance. Moreover, anatomical objects exhibit high variability with respect to their geometry and appearance; large geometrical and photometrical differences exist between different instances of the same structure. Popular methods in medical image shape segmentation use prior knowledge about the object shape



(a) (b)

Figure 1.1: (a) Delineation of anatomical structures in a Posterior Anterior chest radiograph. (b) The cardiothoratic ratio measures the relative heart size and its computation helps to diagnose cardiomegaly.

to be segmented. This expertise is acquired from previously seen objects and expressed through a *model*. The model describes a class of statistical relationships between the image and the shape that is extracted. A set of parameters characterize each member of this class. We refer to *learning* as the task of estimating the parameters that accurately reflect the relationship between a set of training images and a shape model. In *inference* we consider a new image and we rely on the learned model to extract the shape.

In model-based approaches, a common way to represent a shape is through a set of points known as landmarks; for instance, the continuous contours defining a shape can be discretized into a set of sample points. The connectivity between landmarks dictates how to connect the points to form the shape. The task of shape localization is formulated as recovering a set of K anatomical landmarks: $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_K\}$, where every landmark is a 2D position vector $\mathbf{x}_i = (h_i, v_i)$.

1.2 Prior Art

Deformable contour models (DCMs) constitute a main workhorse for detecting shapes from images - starting from the seminal works of Snakes [Kass *et al.* 1987], Deformable Templates [Yuille *et al.* 1992] and Active Shape/Appearance Models [Cootes *et al.* 1998, Cootes *et al.* 1995], DCMs have been thriving in problems involving shapes for more than two decades.

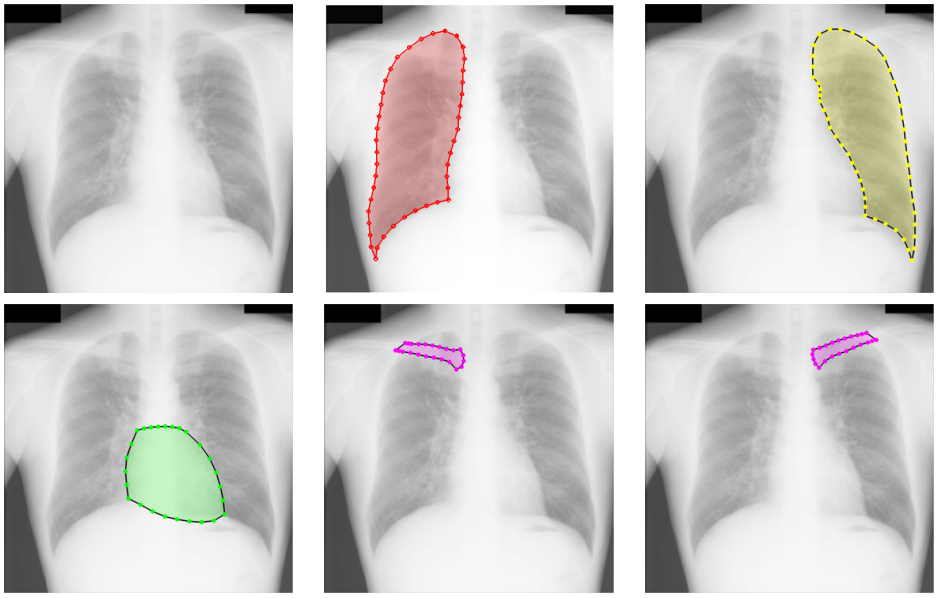
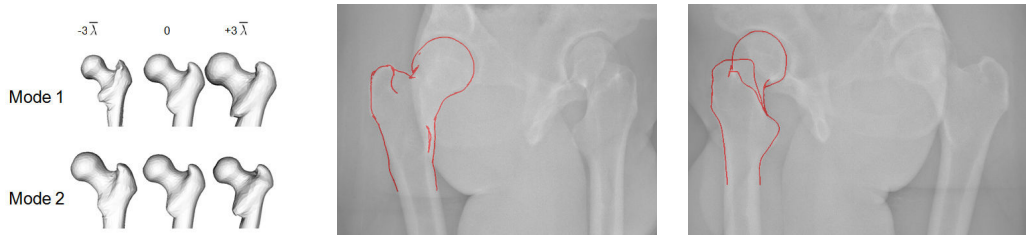


Figure 1.2: Example of shape segmentation in medical image analysis; this is the main application that we consider in this thesis. First image: a Posterior Anterior chest radiograph. Next images: The desired output provided by experts: a set of landmark positions strung together along the contour delineating the shape of an anatomical structure e.g. the right lung, the left lung, the heart, the right clavicle, the left clavicle.

One of the most desirable properties of DCMs is that they allow to cast tasks such as segmentation or tracking in terms of optimization by incorporating the desirable properties of the envisioned solution in the form of a merit function. One can then optimize this function with off-the-shelf techniques, such as Dynamic Programming [Geiger *et al.* 1995], Gradient Descent [Cootes *et al.* 1995], or more dedicated techniques such as curve evolution with Level Sets [Malladi *et al.* 1995] or Finite Element Models [Cohen & Cohen 1993].

Over the previous decades substantial research effort has been devoted to enhancing the geometrical terms in DCMs, including their formulation in intrinsic geometric terms [Caselles *et al.* 1997], the incorporation of more sophisticated contour regularization terms [Kimia *et al.* 2003, Rochery *et al.* 2006, Sundaramoorthi *et al.* 2008] and the introduction of shape priors [Leventon *et al.* 2000, Rousson & Paragios 2002, Cremers 2006, Charpiat *et al.* 2007] in curve evolution.

We can distinguish two broad families of shape models in the modern literature about DCMs, *Global Models* akin to Active Shape Models and Active Appearance Models, and *Local Models* akin to Deformable Part Models. In



a b c

Figure 1.3: An ASM model used in [Boussaid *et al.* 2011] for the proximal femur 3D reconstruction. (a) The first two eigenmodes of variation of ASM. (b) and (c) Segmentation results by projecting the 3D solution on 2D X-ray views.

the following we briefly review these two methods.

1.2.1 Global Models

1.2.1.1 Active Shape Models

Active Shape Models [Cootes & Taylor 1992, Cootes *et al.* 1995] have become increasingly popular since their introduction by [Cootes & Taylor 1992]; several approaches have been proposed to improve their performance [van Ginneken *et al.* 2002, van Assen *et al.* 2003, Langs *et al.* 2006, Li *et al.* 2004, de Bruijne *et al.* 2003, Abi-Nahed *et al.* 2006, Chui & Rangarajan 2003]; in the following, we briefly review the main learning and inference techniques used for ASMs.

Learning

An ASM is composed of two separate models: a global shape model, and a local appearance model. We start by describing learning the global shape model.

We assume that we are provided with a set of training shapes $\mathcal{D} = \{\mathbf{X}_i\}, i = 1 \dots N$. We recall that each shape is represented by a set of K anatomical landmarks: $\mathbf{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_K\}$, where every landmark is a 2D position vector $\mathbf{x}_i = (h_i, v_i)$. We assume as well that the training shapes have been rigidly aligned in advance to the same referential in order to filter out translation, rotation and scale change effects, using e.g. Procrustes analysis. As such shape variability is exclusively due to non-rigid deformations.

ASMs are Statistical models and aim at describing the variation within a class of objects. To this end ASMs assume that the data follows a Gaussian distribution $Pr(\overline{\mathbf{X}}, \Sigma)$. We estimate the mean shape $\overline{\mathbf{X}}$ and the covariance

matrix Σ by using our training set $\mathcal{D}$ and compute:

$$\overline{\mathbf{X}}_{\mathcal{D}} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{X}_i, \tag{1.1}$$

$$\Sigma_{\mathcal{D}} = \frac{1}{N-1}\sum_{i=1}^{N}(\mathbf{X}_i - \overline{\mathbf{X}})(\mathbf{X}_i - \overline{\mathbf{X}})^T. \tag{1.2}$$

Next, ASMs reduce the dimensionality of the data and assume that the data can then be described as lying on the span of a linear subspace. To this end, ASMs apply principal component analysis (PCA) on $\Sigma_{\mathcal{D}}$. This allows to approximate this covariance matrix $\Sigma_{\mathcal{D}}$ through a set of retained eigenvectors $\mathbf{e_i}$ corresponding to largest eigenvalues λ_i. Namely we write:

$$\Sigma_{\mathcal{D}} \approx \mathbf{E}^T \mathrm{diag}(\Lambda)\mathbf{E}, \tag{1.3}$$

where Λ is the vector of retained eigenvalues λ_i and $\mathbf{E}$ is the corresponding set of retained eigenvectors $\mathbf{e_i}$ stored in columns. Hence each shape lying in the subspace of solutions can be written in the new basis composed of the retained eigenvectors as:

$$\mathbf{X} = \overline{\mathbf{X}}_{\mathcal{D}} + \sum_{i=1}^{m}\mathbf{w}_i\mathbf{e_i}, \tag{1.4}$$

where m is set such that a large part of the trace of $\Sigma_{\mathcal{D}}$ is retained. The coordinates of $\mathbf{X}$ in the new basis are $\{\mathbf{w}_i\}, i = 1 \dots m \leq N$ and each coordinate is often constrained to belong to interval $[-3\sqrt{\lambda_i}, 3\sqrt{\lambda_i}]$, where λ_i is the eigenvalue corresponding to the eigenvector $\mathbf{e_i}$. This can be interpreted as rotating the data in the coordinate axes to the directions of maximum variance, retaining only the principal modes of variations of the shape with respect to the mean shape. An example of the principal modes of variation applied in the case of a population of femoral bones -represented by 3D meshes- is shown in Figure 1.3.

The shape model is augmented with knowledge about *the local appearance* around each landmark. An appearance model is associated to each landmark k and learned from annotated data $\mathcal{D}_k = \{I_i, \mathbf{x}_k\}$, where I_i is a training image and $\mathbf{x}_k$ is the location of the landmark. Since ASMs are modular regarding the shape and appearance models, several appearance models were introduced to improve the original appearance model introduced by [Cootes & Taylor 1992]. In the appearance model by [Cootes & Taylor 1992] intensity profiles, centered in the landmark location, and orthogonal to the contour were extracted from training data. Then mean profiles and covariance matrices were then computed. The quality of a new profile was assessed by

the Mahalanobis distance. In [Caselles *et al.* 1997, Rousson & Paragios 2002] profiles of derivatives were used instead and these different profiles were normalized. In [van Assen *et al.* 2003] image patches instead of pixel profiles were employed. Richer landmark- specific local terms were henceforth introduced. Gabor wavelets were used in [Jiao *et al.* 2003] and the resulting feature distributions were modeled using Gaussian mixture models. Steerable features [Freeman & Adelson 1991] were used in [Langs *et al.* 2006] to describe the object appearance. The Active Shape Model with Optimal features (ASMOF) was proposed by [van Ginneken *et al.* 2002]; in this method a given image was fed to a bank of multiscale Gaussian derivative filters. Then, first statistical moments were extracted from local histograms in the filtered images. These moments represented the considered features. An optimal set of features per landmark was then extracted using a forward-backward feature selection scheme.

The advent of machine learning techniques allowed the use of landmark classifers/detectors to express the appearance terms. For instance, a K-nearest neighbors (KNN) classifier was proposed in [de Bruijne *et al.* 2003] to evaluate the probability of a given landmark profile, while the Adaboost [Freund & Schapire 1997] algorithm was used in [Li *et al.* 2004].

Inference

In inference a shape is fitted to a new image by searching for the optimal coordinates $\{\mathbf{w}_i^*\}$ that define the desired shape instance $\mathbf{X}^* = \sum_{i=1}^{m} \mathbf{w}_i^* \mathbf{e_i}$ with respect to image features. The original inference algorithm proposed by [Cootes & Taylor 1992] is an iterative algorithm initialized with the mean shape $\overline{\mathbf{X}}$. At each iteration, the landmark points are updated using the local appearance models and then the shape model is fitted to the updated target points while being regularly projected into the subspace of valid model shapes. This is repeated until a convergence criterion is met. This method can be interpreted a gradient descent and therefore does not guarantee convergence to the globally optimal solution. Other approaches [de Bruijne & Nielsen 2004] reformulate the task as a maximum likelihood problem and optimize it using particle filtering [Isard & Blake 1998]. In [Abi-Nahed *et al.* 2006] a set of candidate points for each landmark is extracted from the image; a point matching algorithm [Chui & Rangarajan 2003] is then used to establish the best correspondences between a legal shape instance and the pool of candidates. In [Boussaid *et al.* 2011] a cost function is formulated based on the geodesic active regions criterion used in [Varshney *et al.* 2009] and is optimized with the Downhill simplex [Lewis *et al.* 2007] method.

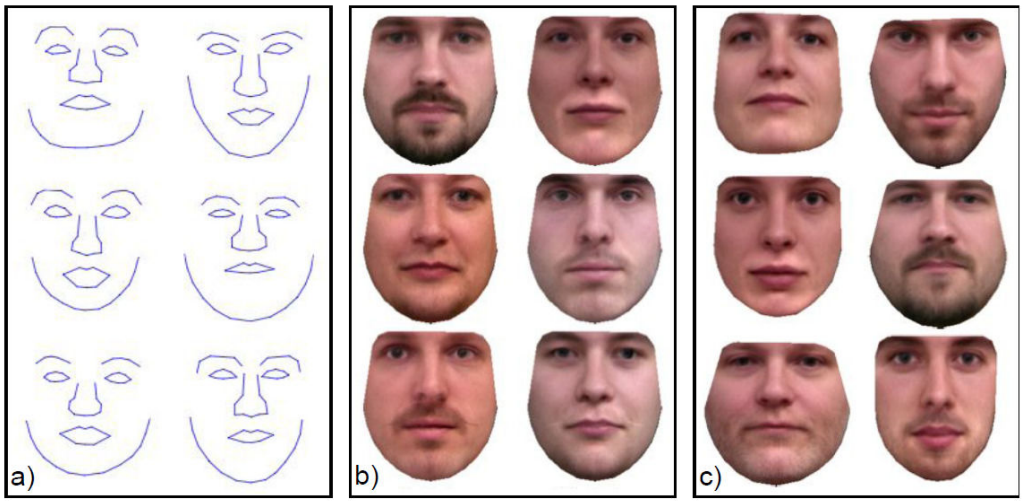


Figure 1.4: Illustration of AAMs from [Prince 2012] adapted from [Stegmann 2002]. a) The shape is parameterized using a subspace model. b) The intensity values for a fixed shape are parameterized using a different subspace model. c) The subspace models are connected in that the weightings of the basis functions in each model are always the same. In this way correlations between shape and texture are described.

1.2.1.2 Active Appearance Models

As with ASMs, an Active Appearance Model [Cootes *et al.* 2001, Torre & Black 2003, Matthews & Baker 2004] is also based on PCA, but builds an integrated model combining shape and appearance. In the following we briefly review the learning and inference techniques proposed for AAMs.

Learning

We assume that we are provided with a set of training images $\mathcal{D} = \{(I_i, \mathbf{X}_i)\}, i = 1 \dots N$. In addition to modeling the shape variations through PCA akin to ASM, AAM models also the texture of the object and its variations. To this end, AAM starts by warping the training images I_i such that the corresponding shape $\mathbf{X}_i$ is transformed onto the mean shape $\overline{\mathbf{X}}$. Each region inside image I_i is represented then by a texture vector. Applying PCA on the texture vectors allows to build an appearance model such that any texture τ can be written as:

$$\tau = \overline{\tau} + \sum_{i=1}^{m'} \mathbf{w}'_i \mathbf{e}'_\mathbf{i}, \tag{1.5}$$

where $\overline{\tau}$ is the mean normalized texture vector, $\mathbf{e}'_\mathbf{i}$ are the eigenvectors and $\mathbf{w}'_i$

are coordinates of the texture τ in this eigenbasis. Each training object can now be described by concatenating in a single vector W_i its shape coordinates in Equation 1.4 and texture coordinates in Equation 1.5. A PCA is applied on W_i to yield an integrated model where the coordinate vector C of an object $\{\mathbf{X}, \tau\}$, controls both shape and texture variations; we write:

$$\mathbf{X} = \overline{\mathbf{X}} + \Phi_{\mathbf{X}}\mathrm{C} \quad (1.6)$$

$$\tau = \overline{\tau} + \Phi_{\tau}\mathrm{C}, \quad (1.7)$$

where $\Phi_{\mathbf{X}}$ and Φ_{τ} are the resulting eigenvectors. To obtain a new synthesized image, the generated texture τ is warped onto the generated geometry $\mathbf{X}$. An example of AAMs application to face objects is shown in Figure 1.4

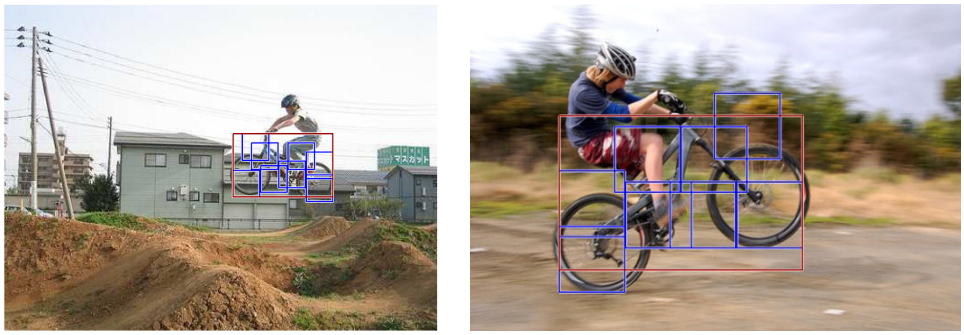
Several variations on the standard AAM scheme training exist. In [Stegmann *et al.* 2001] the intensity information that lies outside the object is added to intensities inside the object when forming the texture vectors. In [van Ginneken *et al.* 2006] intensities sampled along the contour normals at each landmark point are considered in the texture vectors. In [De la Torre & Black 2001], a flexible AAM for faces is built; a face is represented as a set of fixed shape regions which can move independently. AAMs have been besides successfully extended to incorporate 3D information; in [Blanz & Vetter 1999, de La Gorce *et al.* 2011] 3D models of face/hand are trained and used to generate new 2D views. These 3D AAMs have shown ability to fit 2D images and reason about complicated geometric structure using global information about the shape and appearance of the object of interest.

Inference

Given a new image I, the task is to find the optimal parameter vector C such that the synthesized image $I(\mathrm{C})$ resembles the texture image τ_I extracted from image I. We recall that the parameter vector C characterizes a shape $\mathbf{X}_m$ and a texture τ_m. Tis shape $\mathbf{X}_m$ is used to extract the texture τ_I from the image I. To quantify the discrepancy between the two textures, a distance $D(I, \mathrm{C})$ is expressed as:

$$D(I, \mathrm{C}) = \|\tau_m(\mathrm{C}) - \tau_I\|^2 \quad (1.8)$$

The goal is then to search for the parameter vector $\mathrm{C}^\star$ that minimizes this distance. To minimize this function, several approaches -that revolve around gradient descent- have been proposed. [Cootes *et al.* 2001] use training data to acquire prior knowledge about the gradient using numeric differentiation. [Donner *et al.* 2006] use the canonical correlation analysis algorithm instead of numeric differentiation. The authors of [Matthews & Baker 2004] introduce a computationally efficient analytical Quasi-Newton algorithm. Alternative error measurements to Equation 1.8 are also proposed in [Stegmann *et al.* 2001,



(a) (b)

Figure 1.5: Examples of application of DPMs to object detection from [Felzenszwalb *et al.* 2010]; we show an example of detecting an object belonging to 'bike' category. The blue boxes show the part detection results, while the red bounding box localizes the bike in the image.

van Ginneken *et al.* 2006] while an alternative search method is described in [Beichel *et al.* 2005] to improve robustness of AAMs.

1.2.2 Local Models

Currently, Deformable Part Models (DPMs, or 'pictorial structures' [Fischler & Elschlager 1973]) are becoming ubiquitous in computer vision, and are being used in a broad range of high-level tasks, including object detection [Felzenszwalb & Huttenlocher 2005, Felzenszwalb *et al.* 2010], pose estimation [Andriluka *et al.* 2012, Sapp *et al.* 2010, Sapp *et al.* 2011a] and facial landmark localization [Zhu & Ramanan 2012]. DPMs consider an object as a set of rigid components -corresponding to visual parts (e.g eyes, nose, torso, wheel)- that are linked together through non-rigid connections. An object deforms when its parts change their relative positions. Hence an object in an image is represented by parts P_i that are arranged in a deformable configuration.

Each part describes an object's local photometric appearance, and the whole configuration encodes the global geometric layout. We represent a part P_i by its 2D centroid coordinate $\mathbf{x}_i$, where every $\mathbf{x}_i$ is described by a 2D position vector $\mathbf{x}_i = (h_i, v_i)$; we denote vectors with boldface letters and will alternate between the vector notation $\mathbf{x}$ and the horizontal/vertical notation (h, v) based on convenience. A score function $\mathcal{U}_{I,i}(\mathbf{x}_i)$ is used to measures how well a part P_i placed at location $\mathbf{x}_i$ matches to image data.

The geometric layout between parts is expressed by pairwise edges $(P_i, P_j) \in \mathcal{E}$ linking these parts into a global structure, where $\mathcal{E}$ is the set of edges between parts. Pairwise terms $\mathcal{P}_{i,j}(\mathbf{x}_i, \mathbf{x}_j)$ between parts P_i, P_j mea-

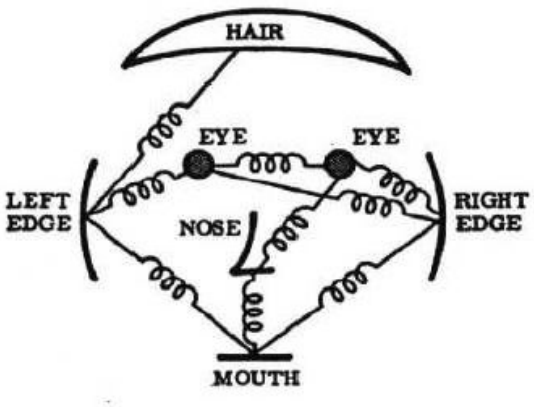


Figure 1.6: A pictorial structure model for a face as introduced by [Fischler & Elschlager 1973]. The mass-spring like terms between visual parts (eyes, noses, mouth, hair) encode geometric constraints between these parts.

sure the deformation between the difference vector $\mathbf{x}_j - \mathbf{x}_j$ and a corresponding precomputed geometrical configuration vector $\mu(P_i, P_j)$ for each $(P_i, P_j) \in \mathcal{E}$, where $\mathcal{E}$ dictates the connectivity between parts.

The location of an object is found in the image by maximizing the corresponding merit function which writes:

$$S_I(\mathbf{X}) = \sum_{i=1}^{K} \mathcal{U}_{I,i}(\mathbf{x}_i) + \sum_{i,j \in \mathcal{E}} \mathcal{P}_{i,j}(\mathbf{x}_i, \mathbf{x}_j), \tag{1.9}$$

where $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_K\}$ is the object represented by the centroids $\mathbf{x}_i$ of its parts P_i.

We now turn to specifying the form of the functions $\mathcal{U}_{I,i}$ and $\mathcal{P}_{i,j}$ and the edge set $\mathcal{E}$ in Equation 1.9.

Unary terms:

The unary terms capture the local fidelity of the image observations at $\mathbf{x}_i$ to a part-specific appearance model. Several works [Felzenszwalb & Huttenlocher 2005, Zhu & Ramanan 2012] in DPMs express this appearance model $\mathcal{U}_i$ in terms of an inner product between a weight vector $\mathbf{u}_i$ and image features extracted around $\mathbf{x}_i$:

$$\mathcal{U}_{I,i}(\mathbf{x}_i) = \langle \mathbf{u}_i, \mathbf{f}_I(\mathbf{x}_i) \rangle, \tag{1.10}$$

where $\mathbf{f}_I(\mathbf{x}_i) : R^2 \rightarrow R^D$ is a mapping from every image coordinates to D-dimensional features; this mapping is defined over image pixel.

One important ingredient to the success of DPMs in computer vision is the use of local feature descriptors to describe the appearance of the model

parts. In order to describe image patches representing the parts -in a manner that is invariant to common transformations (e.g. illumination)- these descriptors rely on histograms of image gradients. A rich line of research has been carried in the computer vision field towards building informative, discriminative and efficient local feature descriptors. These works include HOG [Dalal & Triggs 2005], SIFT [Lowe 2004], SURF [Bay *et al.* 2008], SID [Kokkinos & Yuille 2008], PCA-SIFT [Ke & Sukthankar 2004] and GLOH [Mikolajczyk & Schmid 2005] descriptors.

Many of those descriptors descriptors were first introduced to represent sparse interest points. but recent approaches compute descriptors for every pixel in the image such as Dense SIFT (DSIFT) [Fulkerson *et al.* 2008] and Daisy [Tola *et al.* 2008] and enable therefore their use as a generic low-level image representation on a par with filter banks.

Pairwise terms:

The pairwise term $\mathcal{P}_{i,j}(\mathbf{x}_i, \mathbf{x}_j)$ constrains the location $\mathbf{x}_i = (h_i, v_i)$ of part P_i with respect to its neighbor's location $\mathbf{x}_j = (h_j, v_j)$ with a quadratic expression of the form:

$$\mathcal{P}_{i,j}(\mathbf{x}_i, \mathbf{x}_j) = -\left(\mathbf{x}_j - \mathbf{x}_i - \mu_{i,j}\right)^T C_{i,j} \left(\mathbf{x}_j - \mathbf{x}_i - \mu_{i,j}\right), \tag{1.11}$$

where $C_{i,j} = \mathrm{diag}(\nu_i, \eta_i)$ is the precision matrix and $\mu_{i,j} = (\bar{h}, \bar{v})^T$ is the nominal displacement between $\mathbf{x}_i$ and $\mathbf{x}_j$. The quadratic term in Equation 1.11 is maximal when the displacement between $\mathbf{x}_i, \mathbf{x}_j$ is equal to its nominal value, $\mu_{i,j}$, and decreases for any deviation from it. This can be interpreted as the log-likelihood of a configuration $\mathbf{x}_i, \mathbf{x}_j$ under a Normal distribution with mean $\mu_{i,j}$ and covariance $\Sigma = C_{i,j}^{-1}$. Constraining the concentration matrix to be diagonal, $C_i = \mathrm{diag}(\nu_i, \eta_i)$, allows us to write the pairwise term as a function separable in h and v:

$$\begin{aligned}
\mathcal{P}_{i,j}(\mathbf{x}_i, \mathbf{x}_j) &= -(h_j - h_i - \bar{h})^2 \nu_i - (v_j - v_i - \bar{v})^2 \eta_i. && (1.12)\\
\mathcal{P}_{i,j}(\mathbf{x}_i, \mathbf{x}_j) &= \langle \mathbf{v}_{i,j}, \mathbf{p}(\mathbf{x}_i, \mathbf{x}_j) \rangle, \text{where} && (1.13)\\
\mathbf{v}_{i,j} &= (\nu_i, \eta_j), && (1.14)\\
\mathbf{p}(\mathbf{x}_i, \mathbf{x}_j) &= (-(h_j - h_i - \bar{h})^2, -(v_j - v_i - \bar{v})^2), && (1.15)
\end{aligned}$$

where we can write the pairwise terms as the inner product between a weight and a feature vector.

Graph topology:

In the object detection system with DPMs of [Felzenszwalb *et al.* 2010], the object is modeled through a *star-structured* deformable part model defined by a coarse root component that covers the entire object and higher resolution part components that cover smaller parts of the object. An example for 'bike' object is shown in Figure 1.5. This delivers state-of-the-art

results [Felzenszwalb *et al.* 2010] in object detection challenging benchmarks and become a standard in object recognition research.

For human body pose estimation known -also as articulated models, as shown in Figure 1.7- the used model decomposes a person into body parts (e.g. torso, arms, legs) and the pairwise relations can encode kinematic constraints. The resulting model has a *tree-structure*, as shown in Figure 1.7.

A tree-shaped model is also used for face detection [Felzenszwalb & Huttenlocher 2005]; a root filter captures the face appearance in a coarse resolution while the part filters capture details such as eyes, nose and mouth.

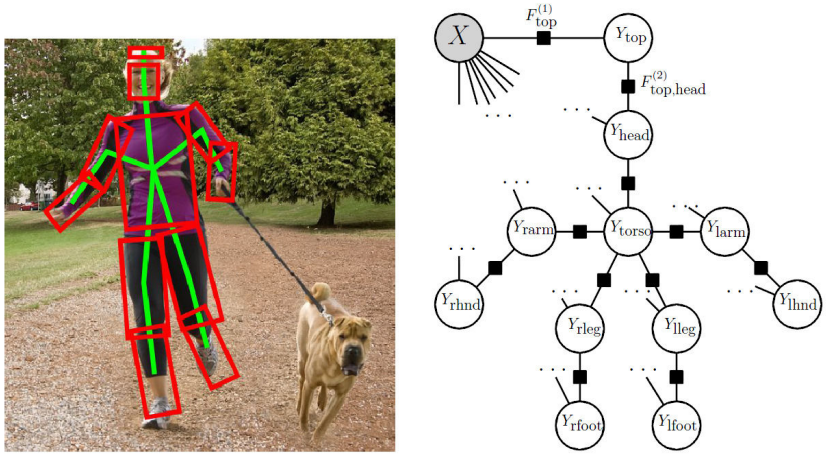


Figure 1.7: Pictorial structures for human body pose estimation. Left: qualitative result produced by pictorial structures from [Felzenszwalb & Huttenlocher 2000]. Right: Pictorial structure model of [Felzenszwalb & Huttenlocher 2000] for pose estimation as illustrated by [Nowozin & Lampert 2011].

Inference

Given an image I, the task is to find the optimal configuration $\mathbf{X}_I^*$ that maximizes Equation 1.9, meaning:

$$\mathbf{X}_I^* = \underset{\mathbf{X}}{\operatorname{argmax}}\, S_I(\mathbf{X}, \mathbf{w}). \tag{1.16}$$

Performing inference efficiently on arbitrary DPMs can be challenging, since this involves a combinatorial problem involving a large label space (consisting of discretized 2D positions). To ensure fast computation, current works [Zhu & Ramanan 2012, Andriluka *et al.* 2012, Sapp *et al.* 2010, Sapp *et al.* 2011a] make the assumption that the structure of the model has

the form of a star -as for object and face detection shown in Figure 1.5- or a tree -as for pose estimation shown in Figure 1.7.

As such, The Max-Product algorithm [Felzenszwalb & Zabih 2011] can be used to recover the globally optimal solution in a time that is quadratic in the number of pixels. Constraining the model furthermore to use separable quadratic pairwise terms allows to couple DP with the Generalized Distance Transforms (GDTs) [Felzenszwalb & Huttenlocher 2004, Felzenszwalb & Huttenlocher 2005]. This reduces the complexity to be linear in the number of pixels.

Learning

Learning in DPMs involves estimating the parameters of each appearance model $\mathcal{U}_{I,i}$ and learning the parameters $C_{i,j}$ and $\mu_{i,j}$ of the pairwise terms $\mathcal{P}_{i,j}$ expressed in Equation 1.11. Other works [Felzenszwalb & Huttenlocher 2005, Besbes & Paragios 2011] aim also at learning the dependencies between parts yielding the structure of the model.

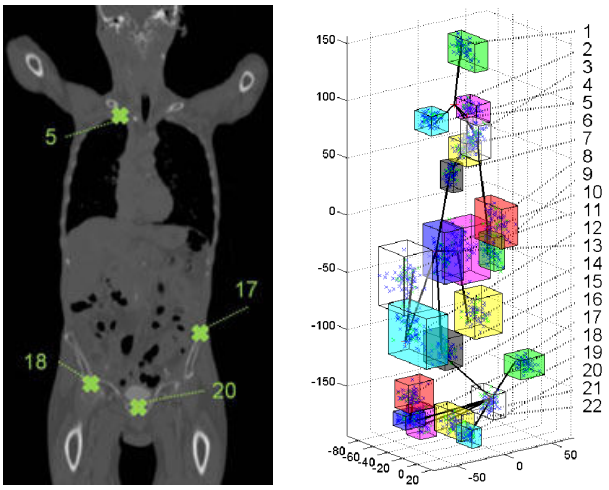
If the appearance models are expressed in terms of an inner product as written in Equation 1.10, a linear classifier can be learned to estimate optimal parameter vector $\mathbf{u}_i$. This is done using linear SVMs in [Dalal & Triggs 2005]. Adaboost is used in [Besbes & Paragios 2011] to learn the appearance models. In [Felzenszwalb & Huttenlocher 2005], a Bayesian formulation of the problem is introduced. A maximum likelihood method is used to learn the appearance parameters as well as the optimal tree structure that best explains the data. In [Barbu & Gramajo 2014] an accurate SVM detector is used and combined with an efficient feature selection method.

The advent of structured prediction learning techniques, allows training all model parameters discriminatively using max-margin Support Vector Machines. In particular, we can see from Equation 1.13 that the pairwise terms are written as the inner product between a weight and a feature vector, and given that the unary terms are also inner products between weights and features, it follows that Equation 1.9 can be written as:

$$S_I(\mathbf{X}, \mathbf{w}) = \langle \mathbf{w}, \mathbf{h}_I(\mathbf{X}) \rangle, \quad \text{where} \tag{1.17}$$

$$\mathbf{w} = (\mathbf{u}_i, \mathbf{v}_{i,j}) \quad \mathbf{h}_I(\mathbf{X}) = (\mathbf{f}_I(\mathbf{x}_i), \mathbf{p}(\mathbf{x}_i, \mathbf{x}_j), \ i, j \in \mathcal{E} \tag{1.18}$$

In the presence of ground truth annotation where the positions of the object parts are known, the training results in a convex optimization problem which can be accurately solved with Structural SVMs as in [Zhu & Ramanan 2012, Andriluka *et al.* 2012, Sapp *et al.* 2010, Sapp *et al.* 2011a]. The Latent SVMs variant aims at training DPMs while the positions of the object parts in the training set are not known as in [Felzenszwalb *et al.* 2010, Kumar *et al.* 2011]. However, the resulting opti-



(a) (b)

Figure 1.8: A DPM as used by [Potesil *et al.* 2010, Potesil *et al.* 2011] for localizing anatomical landmarks in 3D CT volumes. (a) CT of the upper-body. (b) The constraints between parts form a tree-structured graphical model.

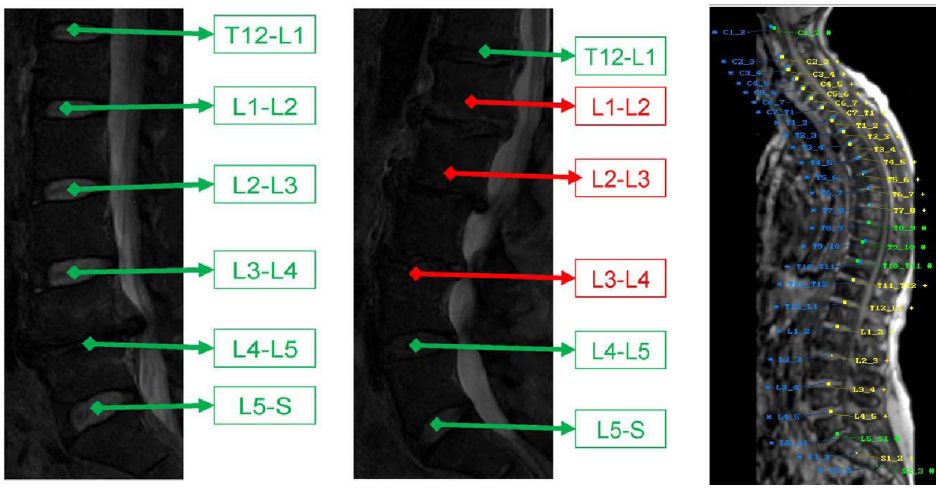
mization problem in not anymore convex and hence the solutions can be only locally optimal.

1.2.2.1 Deformable Part Models in Medical Image Analysis

The success of DPMs in a number of computer vision applications have resulted in their use for organ detection [Potesil *et al.* 2010, Potesil *et al.* 2011, Schmidt *et al.* 2007, Besbes & Paragios 2011, Alomari *et al.* 2011, Potesil *et al.* 2014].

Unlike object detection where deformations are treated as a hurdle, that must be done away with, in order to achieve robust detection, the setting is different in medical imaging where it is typically known a priori that an object (anatomical structure) is present in the image, and the task is to estimate the exact values of the deformation that brings the organs in correspondence with a template. In this case a proper modeling of deformations is needed not to discount variations, but rather to reveal them. This is true also for other tasks where shape modeling is a priority, e.g. in face registration [Saragih & Göcke 2007, Amberg & Vetter 2011] and human pose estimation [Andriluka *et al.* 2012, Sapp *et al.* 2010, Sapp *et al.* 2011a] where accurate body part estimation is the main goal.

In [Potesil *et al.* 2010, Potesil *et al.* 2011] anatomical landmarks in 3D CT volumes are detected in a way similar to body pose estimation as shown in



(a) (b)

Figure 1.9: (a) Results from [Alomari *et al.* 2011]: a DPM is used to localize lumbar discs from MR radiographs. Unsuccessful localization is shown in red. Successful localization is shown in green. (b) Results from [Schmidt *et al.* 2007]: a DPM is used to locate the human vertebral column and to label the intervertebral disks in MR images of total spine.

Figure 1.8; the appearance of landmarks is modeled through the statistics of 3D patchs around them, learned through PCA. The constraints between parts are modeled so as to form a tree-structured graphical model which allows the authors to use dynamic programming for inference. The model involves several free parameters that are set separately by hand.

In [Schmidt *et al.* 2007], intervertebral disks in MR images of total spine are localized. The local part detectors are based on Randomized Classification Trees to provide candidate locations to the inference algorithm. A series of experiments were necessary to estimate the parameters of these classifiers (number of trees, tree depth, number of candidate points) that determine the detection accuracy. Despite the arbitrary structure of the graphical model, the authors apply A^* algorithm on a tree-structured subgraph of the model in order to estimate an upper bound on the optimal solution, which is however not guaranteed to deliver optimal solutions.

The minimal intensity and cost path (MISCP) algorithm [Seghers *et al.* 2007], shown in Figure 1.10 is a similar inference algorithm to circumvent loops in the graph. The authors delineate the shape of anatomical structures in chest radiographs by localizing a set of landmarks strung together to form a closed contour. When it comes to inference, the authors omit one of the problem constraints so as to apply dynamic programming to an open chain structured graph.

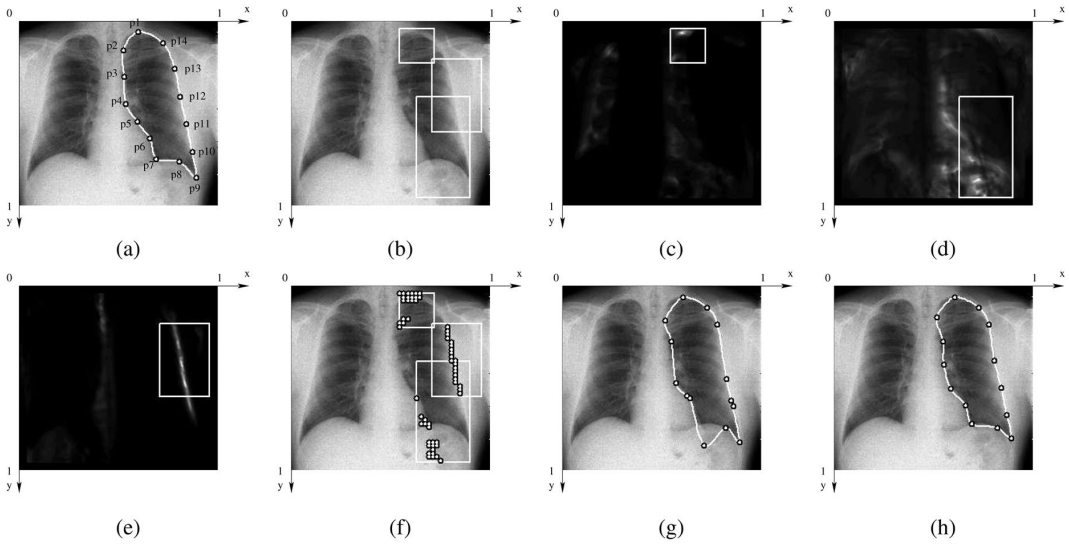


Figure 1.10: Illustration of the MISCP algorithm of [Seghers *et al.* 2007] for the segmentation of an example chest radiograph. A manually delineated left lung with n = 14 landmarks is shown in (a). The search regions for 3 landmarks are shown in (b). Evaluating the unary terms at each pixel results in the score images (c), (d), and (e). The 20 best scoring locations are marked in (f). Results of the minimal cost path fitting with shape knowledge ignored and shape cost incorporated are shown in (g) and (h), respectively.

[Besbes & Paragios 2011] consider the same segmentation problem and build landmark detectors through training an Adaboost classifier for each landmark. They consider convolution-based filters shown in Figure 1.11 as appearance features. The structure of the graph is estimated through manifold learning and unsupervised clustering. The sequential tree-reweighted message passing algorithm (TRW-S) [Kolmogorov 2006] is used for approximate inference.

Overall, we retain the following observations regarding the use of DPMs in medical image analysis: (i) Local feature descriptors (e.g. SIFT, HOG) are not commonly considered as appearance features despite their superiority in performance over convolution based features, as proven in a series of computer vision applications. (ii) The learning of model parameters is done in multiple stages through training point detectors at a first stage, and estimating several parameters separately through trial and error experiments. (iii) Only a subset of candidate solutions are considered -promoted by the point detectors- before running the inference algorithm. (iv) The topology of the model is restricted to a chain- or tree- structured graph to allow the use of DP for inference. (v)

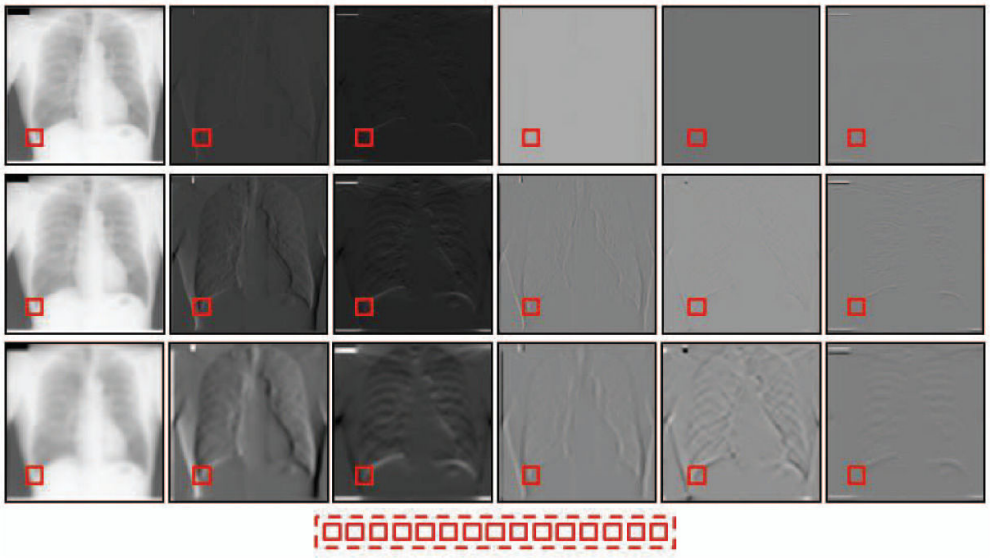


Figure 1.11: Features images in [Besbes & Paragios 2011] using a filter bank. Derivatives up to order 2 of the image are computed after applying Gaussian filters, to form the feature images. Then, image patches (in red) are extracted around a given position to form a feature vector.

The used inference algorithms are relatively slow regarding their complexity in the number of pixels/voxels.

1.2.3 Global vs Local Models

Global models (ASMs/AAMs) exhibit a number of advantages: (i) they are generative in the sense that they allow to synthesize new shapes/images; (ii) they are compact because they describe deformations in terms of few parameters; (iii) they don't need negatives in their training; (iv) AAMs still provide state-of-the-art results for many application such as in [Blanz & Vetter 1999].

However, one of their main shortcomings is that the inference is prone to find a solution which is only locally optimal. Therefore, a sufficiently accurate initialization needs to be provided for the scheme to converge to the correct shape. Due to the use of discrete optimization algorithms such as Max-Product, inference in a number of classes of local models reaches globally optimal solutions, alleviating the need for initialization. Moreover, local models allow to the training in an end-to-end manner as enabled with structured prediction learning, which is not obvious to do with global models.

In [Seghers *et al.* 2007], the authors empirically evaluated the performance of an ASM and a DPM on the same benchmark that we consider in our work,

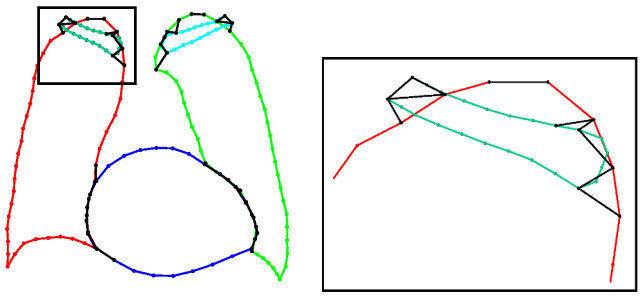


Figure 1.12: Our graphical model's topology reflects the placement of multiple organs corresponding to a patient's heart, lungs, and clavicles. In the detail (right) we are showing in black the edges used to connect the left clavicle and the left lung, as well as the edge that makes the lung contour closed.

and they showed a substantial improvement in performance in favors of DPMs.

1.3 Our Work at a Glance

Our work lies within the scope of local models; we consider DCMs then as a particular instance of DPMs as in [Besbes & Paragios 2011, Seghers *et al.* 2007] to use to find anatomical structures in medical images. Since we use closed contours, by-product this delivers a segmentation of anatomical structures in the medical image. Specifically, we focus on segmentation of lung, heart and clavicle in chest radiographs as shown in Figure 1.2. We use the DPMs machinery to improve segmentation accuracy and assess our results on the publicly available Segmentation in Chest Radiographs (SCR) benchmark [Shiraishi *et al.* 2000].

Our contributions are as follows: in Chapter 2 we revisit training DPMs and present learning techniques to automate the construction of our DCMs using ground-truth annotated data. Specifically, We rely on recent advances on structured prediction learning [Nowozin & Lampert 2011, Joachims *et al.* 2009] to estimate all of our model parameters jointly. Rather that aiming at detection accuracy as in object detection applications, our goal is to produce anatomical structure segmentation as close as possible to the ground truth annotation. To this end, we design a loss function to tune our model's performance according to the criteria used in medical image segmentation. By using the Mean Contour Distance (MCD) as a structured loss during training, we obtain clear test-time performance gains over the standard zero one loss. Training our model with zero-one loss already outperforms the state of the art technique [Seghers *et al.* 2007] and was published in [Boussaid *et al.* 2014]. The use of MCD loss further improves the results

and was published in [Boussaid & Kokkinos 2014]. Furthermore, we explore different options for constructing our model. Namely, we experiment with the gains that we obtain with richer graph topologies that contains loops, different loss functions and with several state-of-the-art dense local features, including Daisy features [Tola *et al.* 2008], dense SIFT features [Fulkerson *et al.* 2008], as well as a multi-scale convolution baseline. We verify that the best results are obtained with rich graph topologies. These advances were only possible because our work on inference, where we deal with efficient inference with loopy graphs.

Chapter 3 handles graphs with arbitrary topologies and proposes an inference algorithm for them. To this end, we decompose the model's graph into a set of open, chain-structured graphs each of which can be efficiently optimized exactly. This results in separate maximization problems but with potential inconsistencies of the individual solutions. We use the *Alternating Direction Method of Multipliers* (ADMM) [Boyd *et al.* 2011, Martins *et al.* 2011a] to fix these potential inconsistencies and show that ADMM yields substantially faster convergence than plain Dual Decomposition-based methods. This inference algorithm was published in [Boussaid & Kokkinos 2014] and is described in Figure 1.14. This algorithm is further accelerated in Chapter 4 by introducing an efficient approach for solving the subproblems that arise in ADMM decomposition.

In Chapter 4 we consider a simple topology for our model corresponding to an open chain-structured graphical model, and we investigate methods to accelerate the resulting inference problem. Our first approach is to implement a Dynamic Programming algorithm accelerated with Generalized Distance Transforms as in [Felzenszwalb & Huttenlocher 2004, Felzenszwalb & Huttenlocher 2005]. This results in a linear time inference algorithm that we consider as our baseline. We then capitalize on recent approaches that solve star-shaped graph optimization in a time practically logarithmic in the number of pixels through a coarse-to-fine approach. We adapt this method to chain structured graphs and integrate it in the Hierarchical A^* Lightest Derivation (HA^*LD) [Felzenszwalb & Mcallester 2007] architecture. This results in an algorithm that is 2^{10} times faster in average than its state-of-the-art counterpart, on the considered database.

All the proposed algorithms are evaluated in a large X-Ray multi-organ image segmentation benchmark, while each of the contributions demonstrate systematic improvements over the current state-of-the-art.

In appendix A, we consider the case of 3D data and we develop an efficient method to find the mode of a 3D kernel density distribution. Our algorithm has guaranteed convergence to the global optimum, and scales logarithmically in the volume size by virtue of recursively subdividing the search space. We

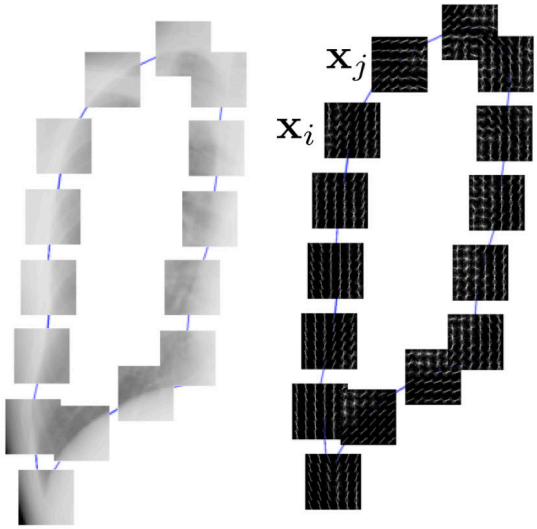


Figure 1.13: Illustration of our DPM for shape based segmentation of the right lung. Our graphical model combines per-landmark local appearance terms and pairwise geometric terms. The unary terms capture the local fidelity of the image features based on histogram of gradients at $\mathbf{x}_i$ to a landmark specific appearance model. The pairwise terms constrains the position of each two consecutive landmarks $\mathbf{x}_i$ and $\mathbf{x}_j$.

use this method to rapidly initialize 3D brain tumor segmentation where we demonstrate substantial acceleration with respect to a standard mean-shift implementation. This work was published in [Boussaid *et al.* 2013].

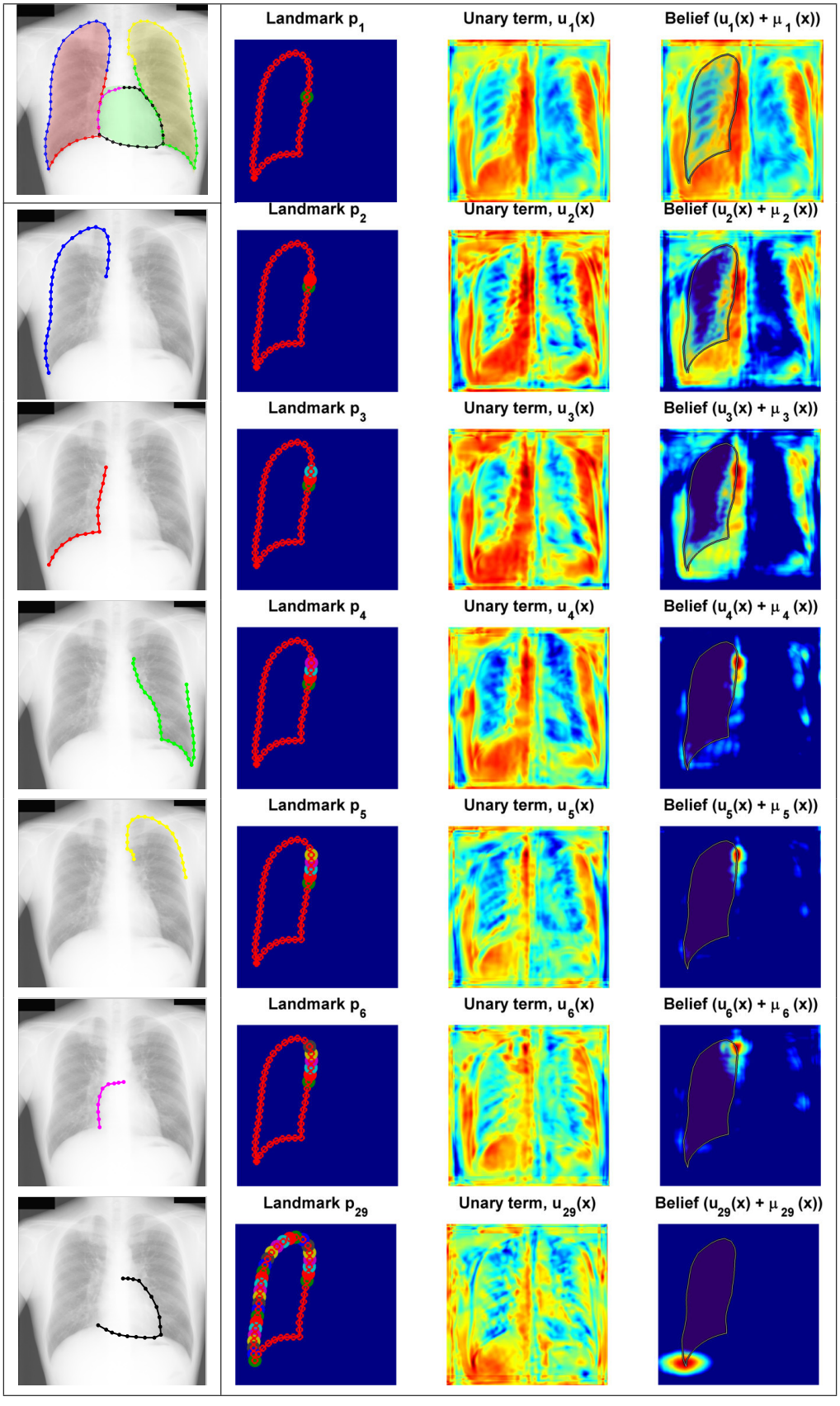


Figure 1.14: Illustration of our inference algorithm in [Boussaid & Kokkinos 2014]. First column: the graph is decomposed into a set of chain structured-subgraphs, each of which is optimized with Dynamic Programming. Second column: inference illustration for the first subgraph; for each landmark, we show its unary term score over pixel locations, and the belief computed for the landmark. The belief is more

Chapter 2

Discriminative Learning of Deformable Contour Models

Contents

2.1 Introduction

In this Chapter we learn the merit function being optimized so as to improve the performance of Deformable Contour Models (DCMs). We use annotated data to estimate the optimal model parameters. This machine learning approach allows us to improve shape localization accuracy in medical images with DCMs.

Our main contribution consists in formulating the task of learning the DCM score function as a large-margin structured prediction problem. Our

algorithm trains DCMs in an joint manner - all the parameters are learned simultaneously. We obtain a training objective that aims at giving the highest score to the ground-truth contour configuration. This effectively shapes our score function so as to place at its optimum the correct contour configurations.

In order to learn the merit function of DCMs in an joint manner, we express it as the inner product of a weight vector with appropriately formed appearance and geometric features and then estimate the optimal weight vector by casting the training problem as structured prediction. Our joint training tackles the estimation of all model parameters in terms of a single objective that directly reflects the performance in the task being solved. In particular, we give as input to our training algorithm, as appearance feature, dense (i.e. computable at every point) image descriptors as opposed to convolution features, which further boost shape detection performance.

In a first shot, we trained with the generic zero-one loss, the resulting model already outperformed the current state-of-the-art [Seghers *et al.* 2007] in the considered benchmark, by virtue of its end-to-end discriminative learning. We improve further the performance of our models through introducing a structured prediction framework suited to the task at hand (i.e shape segmentation). In particular, we use structured SVMs to optimize a loss function specific to contours, considering the minimization of the *mean contour distance* (MCD) performance measure. The resulting learned score function allows us to score each candidate contour according to its MCD to the ground truth configuration, and lends itself to straightforward inclusion into structured prediction learning by virtue of being decomposable into a sum over landmark nodes. We obtain clear test-time performance gains over the model trained with the general zeros-one loss.

We evaluate our method on lung field, heart, and clavicle segmentation tasks using 247 standard posterior-anterior (PA) chest radiographs from the Segmentation in Chest Radiographs (SCR) benchmark. Our learned DCMs systematically outperform the state of the art methods [Seghers *et al.* 2007] according to a host of validation measures including the overlap coefficient, mean contour distance and pixel error rate.

This Chapter is organized as follows: we discuss the choice of appearance features in Section 2.2.1. In Section 2.2.2, we briefly review the two stage learning approach used in current works and consider it as our baseline in our experimental validation. In Section 2.3, we present our approach to learning DCMs. Having setup the machinery to train DCMs, we experiment in Section 2.4 with several options which have a big impact on performance including the choice of the loss function, the topology of the graph and the local appearance features. We demonstrate the merit of our contributions where we show evaluation measures largely superior to the current state-of-the-art.

2.2 Previous Work

2.2.1 Appearance Features

The goal of the unary terms in our models is to localize each landmark based on the image appearance. Using image intensities may be contingent on aspects that do not pertain to the task at hand, e.g. change in contrast, illumination and the particular instance of the object. Image features aim at removing as much of this unwanted variation as possible while retaining the aspects of the image that are critical to the final decision. Therefore, the quality of the image features used to construct the unary terms is a determining factor in performance.

We require that these features should be (a) dense, i.e. computable at every point, to ensure we are not ruling out potential landmark locations and (b) informative and discriminative, so that they can potentially distinguish among different landmarks.

2.2.2 Two-Stage Learning

Current works in medical image analysis estimate the model parameters in a two-stage training manner, using e.g. maximum likelihood (ML) estimation for the pairwise terms, and potentially other combinations of boosting/eigenspaces for the landmark appearance models ([Potesil *et al.* 2010]/[Besbes & Paragios 2011] respectively). In our experience, the resulting unary and pairwise terms can often be incommensurate, and hand-tuning the relative contribution of the resulting terms may be needed. For instance, the minimal intensity and shape cost path (MISCP) algorithm [Seghers *et al.* 2007] uses exhaustive leave-one out experiments to calibrate unary and pairwise terms. In the following, we describe the two stage learning method in more details which we used in [Boussaid *et al.* 2014] and consider it as our baseline method for learning the model parameters.

2.2.2.1 Landmark Detectors

At the first stage, we train a separate classifier $\mathcal{U}_{I,i}$ for each landmark $\mathbf{x}_i$ using the features $\mathbf{f}_I(\mathbf{x}_i)$ that are computed from the image I. These classifiers are learned independently with the goal of localizing the individual node.

Given a set of N training images, where the positions of the landmarks are known, we obtain for each landmark $\mathbf{x}_i$ a set of *positive* examples of size N by extracting features $\mathbf{f}_I(\mathbf{x}_i)$ from each training image around each landmark. We also extract M feature vectors from the background to form a set of *negative*

examples. Then, we obtain the training set:

$$\mathcal{D}_i = \{(\mathbf{f}_I(\mathbf{x}_i)^j, y_i^j)\}, j = 1 \dots P, i = 1 \dots K \tag{2.1}$$

where $P = N + M$ is the number of examples and the class label $y_i^j \in \{-1, 1\}$ refers to the landmark/background. We train the classifiers $g_i(\mathbf{f}_I(\mathbf{x}_i))$ in order to discriminate the landmarks from their neighborhoods. Hence, the unary term $\mathcal{U}_{I,i}(\mathbf{x}_i)$ is the response of the ith classifier $g_i(\mathbf{f}_I(\mathbf{x}_i))$ with respect to the feature vector $\mathbf{f}_I(\mathbf{x}_i)$. Each classifier $g_i(\mathbf{f}_I(\mathbf{x}_i))$ depends on a parameter vector $\mathbf{u}_i$. This forms a classification problem. In order to estimate the parameter vector $\mathbf{u}_i$, the classifier can be trained with Adaboost or linear SVMs or Logistic Regression. In this work we use linear SVMs. Therefore, our unary term $\mathcal{U}_{I,i}(\mathbf{x}_i)$ can be written as:

$$\mathcal{U}_{I,i}(\mathbf{x}_i) = \langle \mathbf{u}_i, \mathbf{f}_I(\mathbf{x}_i) \rangle. \tag{2.2}$$

2.2.2.2 Pairwise Term Parameter Estimation

Now we turn to estimate the parameters that appear in the pairwise terms $\mathcal{P}_{i,j}(\mathbf{x}_i, \mathbf{x}_j)$. We recall that our pairwise terms write:

$$\mathcal{P}_{i,j}(\mathbf{x}_i, \mathbf{x}_j) = -(h_j - h_i - \bar{h})^2 \nu_i - (v_j - v_i - \bar{v})^2 \eta_i. \tag{2.3}$$

Under the assumption that each vector connecting two neighboring landmarks $\mathbf{x}_i, \mathbf{x}_j$ is normally distributed, we estimate its mean $\mu_{i,j}$ and covariance $\Sigma_{i,j}$ from the training shapes. The pairwise term $\mathcal{P}_{i,j}$ of an observed configuration $\mathbf{x}_i, \mathbf{x}_j$ can then be seen as a log-likelihood under the Normal distribution $\mathcal{N}(\mu_{i,j}, \Sigma_{i,j})$.

2.2.2.3 Calibration

Estimating each subset of the model parameters separately may not result in a coherent model and is prone to a poor performance. To circumvent this problem, a second stage of learning is needed. In this stage the magnitude of importance of the independent detectors and the likelihood terms can be adjusted. The merit function is then formulated as follows

$$S_I(\mathbf{X}) = \sum_{i=1}^{K} \alpha_i \mathcal{U}_{I,i}(\mathbf{x}_i) + \sum_{i,j \in \mathcal{E}} \beta_{i,j} \mathcal{P}_{i,j}(\mathbf{x}_i, \mathbf{x}_j), \tag{2.4}$$

where the task is to find the optimal weighting scalars $\{\alpha_i, i = \dots K\}$ and $\{\beta_{i,j}, i, j \in \mathcal{E}\}$. Another more simplistic variant is expressed as:

$$S_I(\mathbf{X}) = \sum_{i=1}^{K} \mathcal{U}_{I,i}(\mathbf{x}_i) + \lambda \sum_{i,j \in \mathcal{E}} \mathcal{P}_{i,j}(\mathbf{x}_i, \mathbf{x}_j), \tag{2.5}$$

where the task is to find λ that calibrates to ensure a trade-off between unary terms and pairwise terms.

In practice, these parameters are frequently hand-tuned or estimated by means of trial and error, or using basic machine learning techniques such as cross validation or leave one out experiments.

2.3 Structured Prediction Learning

2.3.1 Merit Function Formulation

We firstly recall that we can see our pairwise terms as the inner product between a weight and a feature vector.

$$\begin{aligned} \mathcal{P}_{i,j}(\mathbf{x}_i, \mathbf{x}_j) &= \langle \mathbf{v}_{i,j}, \mathbf{p}(\mathbf{x}_i, \mathbf{x}_j) \rangle, \text{where} && (2.6) \\ \mathbf{v}_{i,j} &= (\nu_i, \eta_j), && (2.7) \\ \mathbf{p}(\mathbf{x}_i, \mathbf{x}_j) &= (-(h_j - h_i - \bar{h})^2, -(v_j - v_i - \bar{v})^2). && (2.8) \end{aligned}$$

Since, as per Equation 2.2, the unary terms are also inner products between weights and features and our merit function is additive, it follows that our merit function is the inner product between two vectors:

$$\begin{aligned} S_I(\mathbf{X}) &= \langle \mathbf{w}, \mathbf{h}_I(\mathbf{X}) \rangle \quad \text{where} && (2.9) \\ \mathbf{w} &= (\mathbf{u}_1, \ldots, \mathbf{u}_K, \mathbf{v}_1, \ldots, \mathbf{v}_{K-1}) && (2.10) \\ \mathbf{h}_I(\mathbf{X}) &= (\mathbf{f}_I(\mathbf{x}_1), \ldots, \mathbf{f}_I(\mathbf{x}_K), \mathbf{p}(\mathbf{x}_1, \mathbf{x}_2), \ldots, \mathbf{p}(\mathbf{x}_{K-1}, \mathbf{x}_K)) && (2.11) \end{aligned}$$

To make the dependence of $S_I(\mathbf{X})$ on $\mathbf{w}$ explicit, we will be denoting the score function as $S_I(\mathbf{X}, \mathbf{w})$ henceforth.

2.3.2 Structured Prediction

We assume that we have been provided with a training set of images and associated ground-truth contour locations, which we will denote as

$$\mathcal{D} = \{(I_i, \hat{\mathbf{X}}_i)\}, i = 1 \ldots N. \tag{2.12}$$

Our goal is to use this training set to learn a merit function such that on new, unseen, images the optimal contour configuration will be close to the respective ground truth configuration. More specifically, we can see the inference problem that will be treated in Chapter 3 as defining a mapping $\Gamma : \mathcal{I} \rightarrow \mathcal{X}$ between the space of images and the space of contours:

$$\Gamma_{\mathbf{w}}[I] = \mathbf{X}_I^* \tag{2.13}$$

Namely, given a parameter vector $\mathbf{w}$ we have a mapping ('operator') that takes an image as input and outputs a contour. Our goal is to estimate $\mathbf{w}$ so that this mapping will deliver contours close to the desired ones. Unlike the standard binary classification problem, where the desired output of a mapping is a binary, or discrete, label, here we face a problem with structured outputs.

The solution of such *structured prediction* problems has been recently addressed in the machine learning community [Joachims *et al.* 2009] and has delivered fruitful results in a host of computer vision problems [Nowozin & Lampert 2011] and in pose estimation [Sapp *et al.* 2011b, Mittal *et al.* 2012] in particular. We refer to the references above for a more thorough treatment of structured prediction and proceed to a presentation taylored to our case, which allows us to simplify the presentation.

2.3.3 Learning the DCM merit function

We consider that we have a loss function $\Delta : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}_+$, where $\Delta(\mathbf{X}_1, \mathbf{X}_2)$ indicates the discrepancy between two elements of the output space. We will use $\Delta(\mathbf{X}_I, \hat{\mathbf{X}}_I)$ to specify the cost of predicting a shape $\mathbf{X}_I^\star = \text{argmax}_{\mathbf{X}\in\mathcal{X}} S_{I_i}(\mathbf{X}, \mathbf{w})$ for an image I when the correct shape is $\hat{\mathbf{X}}_I$. By definition, we have $\Delta(\hat{\mathbf{X}}_I, \hat{\mathbf{X}}_I) = 0$.

Learning in these settings amounts to choosing $\mathbf{w}$ such that the total loss on all training instances in $\mathcal{D}$ is minimized and adding to it a regularizer. Formally, we write:

$$\mathbf{w}^\star = \underset{\mathbf{w}}{\text{argmin}} \|\mathbf{w}\|_2^2 + \sum_{i=1}^{N} \Delta(\hat{\mathbf{X}}_{I_i}, \mathbf{X}_i^\star) \tag{2.14}$$

$$= \underset{\mathbf{w}}{\text{argmin}} \|\mathbf{w}\|_2^2 + \sum_{i=1}^{N} \Delta(\hat{\mathbf{X}}_{I_i}, \underset{\mathbf{X}\in\mathcal{X}}{\text{argmax}}\, S_{I_i}(\mathbf{X}, \mathbf{w})) \tag{2.15}$$

To guarantee a good generalization on new unseen images, $\mathbf{w}$ is constrained to be sufficiently smooth by minimizing its squared L_2 norm.

The resulting optimization problem is hard to solve since the objective function is non convex in w. Nonetheless, we still can achieve a good prediction accuracy by minimizing a convex upper bound to 2.15. An upper bound to $\Delta(\hat{\mathbf{X}}_{I_i}, \mathbf{X}_i^*)$ can be found as :

$$\Delta(\hat{\mathbf{X}}_{I_i}, \mathbf{X}_i^*) \leq \Delta(\hat{\mathbf{X}}_{I_i}, \mathbf{X}_i^*) + S_{I_i}(\mathbf{X}^*, \mathbf{w}) - S_{I_i}(\hat{\mathbf{X}}_i, \mathbf{w}) \tag{2.16}$$

$$\leq \max_{\mathbf{X}\in\mathcal{X}} \Delta(\hat{\mathbf{X}}_{I_i}, \mathbf{X}) + S_{I_i}(\mathbf{X}, \mathbf{w}) - S_{I_i}(\hat{\mathbf{X}}_i, \mathbf{w}) \tag{2.17}$$

The first inequality 2.16 holds because $\mathbf{X}^*$ has the maximum score over all contours including $\hat{\mathbf{X}}_i$. In 2.17, we replaced the loss augmented score of $\mathbf{X}^*$,

$\Delta(\hat{\mathbf{X}}_{I_i}, \mathbf{X}_i^*) + S_{I_i}(\mathbf{X}^*, \mathbf{w})$ with its maximum value $\Delta(\hat{\mathbf{X}}_{I_i}, \mathbf{X}_i^*) + S_{I_i}(\mathbf{X}^*, \mathbf{w})$ over all possible contours $\mathbf{X} \in \mathcal{X}$.

We can interpret minimizing this upper bound as requiring that for a training image I_i any configuration $\mathbf{X}$ other than the ground truth $\hat{\mathbf{X}}_i$ should score below $\hat{\mathbf{X}}_i$ by a certain margin; we can write this requirement concisely as:

$$S_{I_i}(\hat{\mathbf{X}}_i, \mathbf{w}) \geq S_{I_i}(\mathbf{X}, \mathbf{w}) + \Delta(\mathbf{X}, \hat{\mathbf{X}}_i), \quad \forall \mathbf{X} \tag{2.18}$$

In particular for $\mathbf{X} = \hat{\mathbf{X}}_i$ the loss on the right hand side vanishes, and the inequality is satisfied as an equality. Otherwise the inequality requires $s_{I_i}(\hat{\mathbf{X}}_i, \mathbf{w})$ to be larger than any other $s_{I_i}(\mathbf{X}, \mathbf{w})$ by at least $\Delta(\mathbf{X}, \hat{\mathbf{X}}_i)$. We can see this requirement as imposing an ordering in the space of contours, such that the ground truth contour has the highest rank, and with a margin from the second-best contour. For certain cases meeting this set of constraints may not be feasible; we therefore introduce a slack variable ξ_i associated with the i-th training example:

$$\begin{aligned} S_{I_i}(\hat{\mathbf{X}}_i, \mathbf{w}) + \xi_i &\geq S_{I_i}(\mathbf{X}, \mathbf{w}) + \Delta(\mathbf{X}, \hat{\mathbf{X}}_i), \quad \forall \mathbf{X} && (2.19) \\ \xi_i &\geq 0 && (2.20) \end{aligned}$$

which relaxes the set of constraints; namely we 'push' the score of $s_{I_i}(\hat{\mathbf{X}}_i, \mathbf{w})$ upwards by ξ_i to make the set of constraints satisfiable. To avoid this relaxation in cases where it is unnecessary, we penalize the sum of slack variables through our training criterion. In particular we cast training our merit function as the optimization of the following quadratic program (QP):

$$\begin{aligned} \text{minimize} \quad & C(\mathbf{w}, \xi) = \|\mathbf{w}\|_2^2 + \lambda \sum_{i=1}^{N} \xi_i && (2.21) \\ \text{subject to} \quad & S_{I_i}(\hat{\mathbf{X}}_i, \mathbf{w}) - S_{I_i}(\mathbf{X}, \mathbf{w}) \geq \Delta(\mathbf{X}, \hat{\mathbf{X}}_i) - \xi_i, \quad \forall \mathbf{X}, i && (2.22) \\ & \xi_i \geq 0, \quad \forall i && (2.23) \end{aligned}$$

The first term in the training criterion regularizes the solution and guarantees good generalization, while the second term penalizes the amount by which the constraints are relaxed; the set of constraints in Equation 2.22 requires that we rank the ground truth configuration higher than all alternatives for a given image. The cost function is quadratic in $\mathbf{w}$ and linear in ξ, while the set of constraints is linear in $\mathbf{w}$, since $S_{I_i}(\mathbf{X}, \mathbf{w})$, $S_{I_i}(\mathbf{X}, \hat{\mathbf{X}}_i)$ are linear in $\mathbf{w}$; as such, a single global optimum exists and can be found in principle with any QP solver.

In practice solving this problem can be computationally challenging due to the large number of constraints implied by considering all possible contours $\mathbf{X}$. For that purpose we use cutting plane optimization which solves the

QP iteratively by appending at each iteration the most violated constraint [Joachims *et al.* 2009]; i.e. for every image i we solve the optimization problem, known as the augmented inference problem:

$$\mathbf{X}_{cp}^{i} = \underset{\mathbf{X}}{\operatorname{argmax}}\, S_{I_i}(\mathbf{X}, \mathbf{w}) - S_{I_i}(\hat{\mathbf{X}}_i, \mathbf{w}) + \Delta(\mathbf{X}, \hat{\mathbf{X}}_i) \tag{2.24}$$

This implies that we need to, given the current value of $\mathbf{w}$, find a $\mathbf{X}_{cp}^{i}$ that has good score according to the model, and a high loss according to the ground-truth. The $-S_{I_i}(\hat{\mathbf{X}}_i, \mathbf{w})$ term corresponding to the ground truth score is constant with respect to $\mathbf{X}$, and is therefore omitted. We then append the resulting constraint to the set of constraints already entertained.

In order to accelerate the convergence of the algorithm, [Joachims *et al.* 2009] formulate the problem with only one slack variable ξ as follows:

$$\text{minimize} \quad C(\mathbf{w}, \xi) = \|\mathbf{w}\|_2^2 + \lambda\xi \tag{2.25}$$

$$\text{subject to}$$

$$S_{I_i}(\hat{\mathbf{X}}_i, \mathbf{w}) - S_{I_i}(\mathbf{X}, \mathbf{w}) \geq \Delta(\mathbf{X}, \hat{\mathbf{X}}_i) - \xi, \forall \mathbf{X}, i \tag{2.26}$$

$$\xi \geq 0,$$

This is the standard way of training a structured output SVM along the lines of [Joachims *et al.* 2009]. One last detail is that the parameters of the pairwise terms in Equation 1.15 need to remain positive, since otherwise they will reward deviations from the nominal displacements; we add these positivity constraints to the set of constraints. We make sure that these hard constraints are never violated through the iterations of the algorithm.

2.3.4 Structured Prediction for Segmentation

In this section we discuss the design of appropriate loss functions. A loss function allows us to measure the performance of a particular weight vector in terms of the loss $\Delta(\mathbf{X}_{I_i}^{*}, \hat{\mathbf{X}}_i)$ which represents the cost incurred by labelling image i as $\mathbf{X}_{I_i}^{*}$ when the ground truth is $\hat{\mathbf{X}}_i$.

The simplest option we consider is the general zero-one loss:

$$\Delta_{0-1}(\mathbf{X}, \hat{\mathbf{X}}) = \begin{cases} 0, & \mathbf{X} = \hat{\mathbf{X}} \\ 1, & \text{otherwise,} \end{cases} \tag{2.27}$$

which penalizes any discrepancy between the ground truth and the recovered solution. For the zero-one loss the augmented inference problem in equation 2.24 boils down to recovering any optimum of $s_{I_i}(\mathbf{X}, \mathbf{w})$ different to $\hat{\mathbf{X}}$.

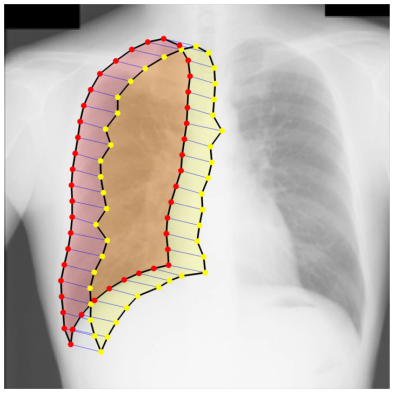


Figure 2.1: The Mean Contour Distance (MCD) measures the average distance of the landmarks of two contours.

Different loss functions can be used however to better reflect the nature of our problem. In this work, we aim at directly optimizing a performance measure specific to the problem at hand. In particular we use the Mean Contour Distance (MCD) which measures the average distance of the landmarks of two contours. In our case, the contours are discretized in a set of landmark positions, connected through straight lines. The MCD between two contours $\mathbf{X}$ and $\hat{\mathbf{X}}$ is then defined as:

$$\Delta_{mcd}(\mathbf{X}, \hat{\mathbf{X}}) = \frac{1}{K} \sum_{i=1}^{K} ||\mathbf{x}_i - \hat{\mathbf{x}}_i||_2 \,. \tag{2.28}$$

The resulting augmented inference problem can be written as:

$$\mathbf{X}^i_{cp} = \underset{\mathbf{X}}{\operatorname{argmax}} \sum_{k=1}^{K} (\mathcal{U}_{I_i,k}(\mathbf{x}_k) + \delta(\mathbf{x}_k, \hat{\mathbf{x}}_k)) + \sum_{(k,j)\in\mathcal{E}} \mathcal{P}_k(\mathbf{x}_k, \mathbf{x}_j),$$

where $\delta(\mathbf{x}_i, \hat{\mathbf{x}}_i) = \frac{1}{K} ||\mathbf{x}_i - \hat{\mathbf{x}}_i||_2$ is the per-landmark decomposition of the loss; since this term is absorbed in the unary term, it follows that optimizing this last expression can be done as efficiently as solving the original optimization problem.

2.4 Experimental Evaluation

2.4.1 The dataset

We systematically evaluate our method on the publicly available dataset of [Shiraishi *et al.* 2000, van Ginneken *et al.* 2006] which contains 247 standard posterior anterior chest radiographs of healthy and non-healthy subjects (presenting nodules). The database contains segmentations from radiologists,

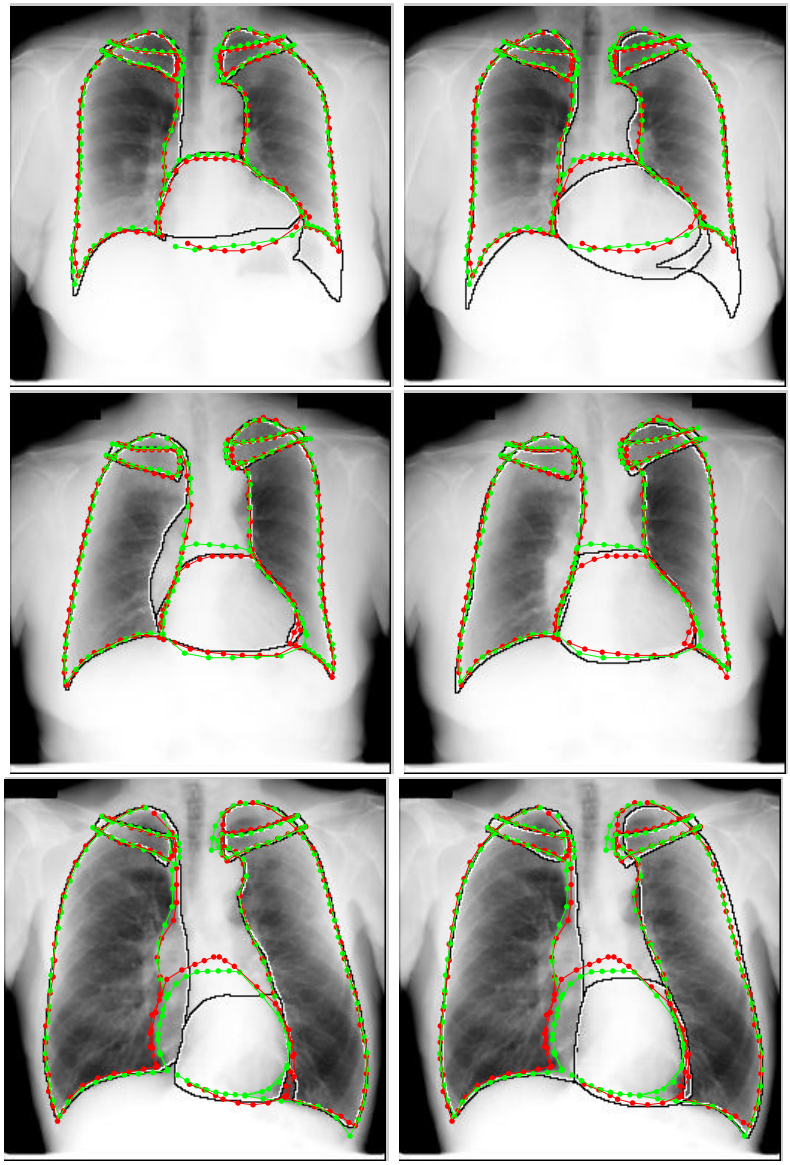


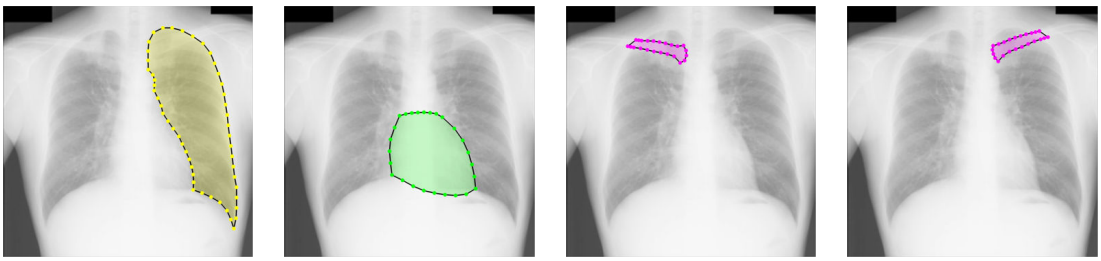
Figure 2.2: Left: Our segmentations (red) superimposed on the results of the MISCP algorithm [Seghers *et al.* 2007] (black contours). Right: Our segmentations (red) superimposed on the results of ASM based method of [van Ginneken *et al.* 2006] (black contours). The ground truth segmentations are shown in green. Each row represents the same patient chest.

Table 2.1: Lung segmentation performance measures including Dice and Jaccard coefficients (larger is better) and Means Contour Distance (smaller is better). We compare the performance of the previous state-of-the-art, MISCP [Seghers *et al.* 2007], ASM [van Ginneken *et al.* 2006], and different choices for our method, involving dense SIFT at a resolution of 4 pixels per bin and a convolution baseline (CONV) with steerable-scalable filters. The suffix TS indicates two-stage and JT indicates joint training. The proposed method corresponds to the use of SIFT descriptors and joint training

	Right Lung (44 landmarks)			Left Lung (50 landmarks)		
method	Dice	Jaccard	M.C.D	Dice	Jaccard	M.C.D
Proposed method	**97.85**	**95.8**	**1.82**	**97.52**	**95.2**	**1.96**
SIFT+TS	97.17	94.6	1.96	96.65	93.6	2.52
CONV+JT	97.26	94.5	1.82	96.8	93.8	2.66
CONV+TS	96.84	93.9	1.96	95.81	92.0	2.38
[Seghers *et al.* 2007]	N/A	94.0	2.1	N/A	92.0	2.38
[van Ginneken *et al.* 2006]	N/A	92.1	2.66	N/A	88.6	3.78

which provide a delineation of the lung fields, the heart and the clavicles as shown in Figure 2.4.1. Gold standard segmentation masks are hence available as well as corresponding landmark positions lying on the contour.

Following the evaluation setup described in [Shiraishi *et al.* 2000], we use 123 images for training and a separate set of 124 images for testing, using the provided training/testing split; all of the reported results are on the whole test set, using images of size 256 by 256.



Left lung Heart Right Clavicle Left Clavicle

Figure 2.3: We consider the publicly available SCR dataset to assess the performance of our method. This dataset contains along with chest radiographs, manual segmentations of the right lung, the left lung, the heart, the right clavicle and the left clavicle.

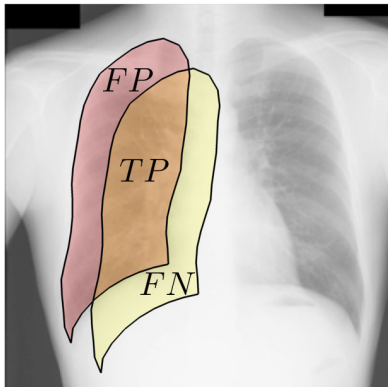


Figure 2.4: Illustration of the true positives TP, true negatives TN and false postives FP involved in the Dice and Jaccard evaluation measure computation.

2.4.2 Evaluation Methodology

We evaluate the performance of our method by comparing our segmentations to the ground truth segmentations by means of a range of validation measures; these include Jaccard, Dice coefficients, mean contour distance (MCD) and pixel error measures. The Jaccard coefficient between two regions $\mathcal{L}$ and $\hat{\mathcal{L}}$ is defined by:

$$J(\hat{\mathcal{L}}, \mathcal{L}) = \frac{TP}{TP + FN + FP}, \tag{2.29}$$

where the true positive (TP) area is the area correctly classified as object, the false positive (FP) area is the area incorrectly classified as object and the false negative (FN) area is the area incorrectly classified as background. This amounts to computing the area of intersection divided by the area of union of two regions:

$$J(\hat{\mathcal{L}}, \mathcal{L}) = \frac{\left|\mathcal{L} \cap \hat{\mathcal{L}}\right|}{\left|\mathcal{L} \cup \hat{\mathcal{L}}\right|}. \tag{2.30}$$

The Dice coefficient between two regions is defined as two times the area of their intersection divided by the sum of their areas:

$$D(\hat{\mathcal{L}}, \mathcal{L}) = \frac{2\left|\mathcal{L} \cap \hat{\mathcal{L}}\right|}{|\mathcal{L}| + \left|\hat{\mathcal{L}}\right|}. \tag{2.31}$$

The Dice coefficient is an equivalent measure to Jaccard index. We include it for completeness. These overlap coefficients are high for large, simple objects and low for complex, small objects.

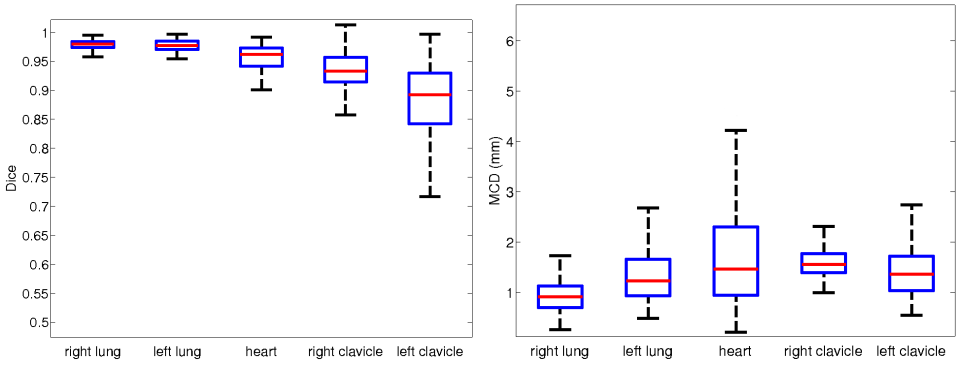


Figure 2.5: Dice coefficients (left) and Mean Contour Distance statistics (right) on different chest organs (the overall decrease in the DICE coefficients for the clavicles is anticipated due to their smaller scale).

Table 2.2: Heart and clavicle segmentation performance measures including Dice and Jaccard coefficients (larger is better) and means contour distance (smaller is better). We compare the performance of the previous state-of-the-art, ASM [van Ginneken *et al.* 2006], and our best-performing method involving dense SIFT at a resolution of 4 pixels per bin and joint training.

	Heart (26 landmarks)			Clavicles (23 landmarks)		
method	Dice	Jaccard	M.C.D	Dice	Jaccard	M.C.D
Proposed method	**94.8**	**90.1**	**3.03**	**90.4**	**80.1**	**1.4**
[van Ginneken *et al.* 2006]	N/A	81.4	5.96	N/A	73.4	2.04

The MCD between two contours A and B is defined in terms of the average distance from a point on the contour A to the nearest point on the contour B. It is obtained by averaging the distance A B to the distance B A. All the reported MCD measures are in millimeter. The pixel error index is defined as the proportion of pixels for which any of the five object labels (lungs, heart, clavicles) is not in agreement with the ground truth segmentation.

To validate the merit of our contributions, we assess the performance of our work against existing baselines. We note that both optimization and learning do not suffer from local minima issues, while the complexity penalty coefficient of structured SVM is determined with 10-fold cross validation; we can thus attribute any difference in the final results exclusively to the low-level feature, model structure, and loss function choices.

2.4.3 Design choices

2.4.3.1 Convolution-based vs Descriptor-based Appearance Features

Existing works in medical imaging either use potentially rich, but sparse features [Schmidt *et al.* 2007, Besbes & Paragios 2011] -which requires recovering from detector failures- or dense, but less discriminative features using convolution with filterbanks [Potesil *et al.* 2010]. Most of these works rely on per-pixel image transformation (e.g. convolutions) and develop features based on steerable filters [Freeman & Adelson 1991] and Gabor filters [Wang *et al.* 2010a]. Recent approaches introduced descriptors to be used as image features (e.g. [Tola *et al.* 2008]). The goal of a descriptor is to provide a compact representations that summarize the contents of an image region or a patch around a pixel.

Scale-Invariant Feature Transforms (SIFT) or Histograms-of-Gradients (HOG) describe shape around a point as a distribution on invariant features, such as a histogram of gradients. Several works have used sparse, descriptor-based techniques to construct a shortlist of candidate part locations [Toews & Wells III 2012, Schmidt *et al.* 2007, Besbes & Paragios 2011], but more recently dense features have been proposed in [Potesil *et al.* 2010, Potesil *et al.* 2011]. In this work we use dense, informative features, including Daisy descriptors [Tola *et al.* 2008] and dense SIFT [Fulkerson *et al.* 2008]. In the following we briefly describe these two 'dense' descriptors since they form a relevant component of our work.

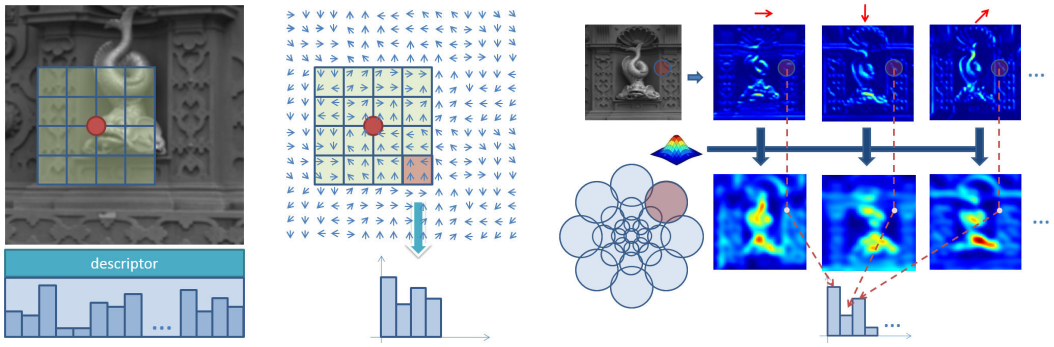
Dense SIFT Descriptor:

Given an image patch with a selected scale and orientation, the Scale Invariant Feature Transform (SIFT) [Lowe 2004] is built by aggregating oriented gradients over three dimensions: the spatial coordinates and the gradient orientation. The resulting histograms of gradients are then concatenated. This illustrated in Figure 2.6 (a). Dense Sift [Fulkerson *et al.* 2008] allows the computation of the descriptor around all points in the image domain, for constant scales and orientations.

DAISY Descriptor:

Daisy descriptor was introduced to design a fast descriptor for dense computation. Daisy aggregates image gradients obtained from convolutions over gradient images to built its histograms. The convolution operation is separable and fast to compute. An illustration of daisy computation is provided in Figure 2.6 (b)

Both of these descriptors can be efficiently computed in batch mode and as such can be used on a par with standard filterbank features. But they come with improvements in point matching/classification performance by virtue of



(a) (b)

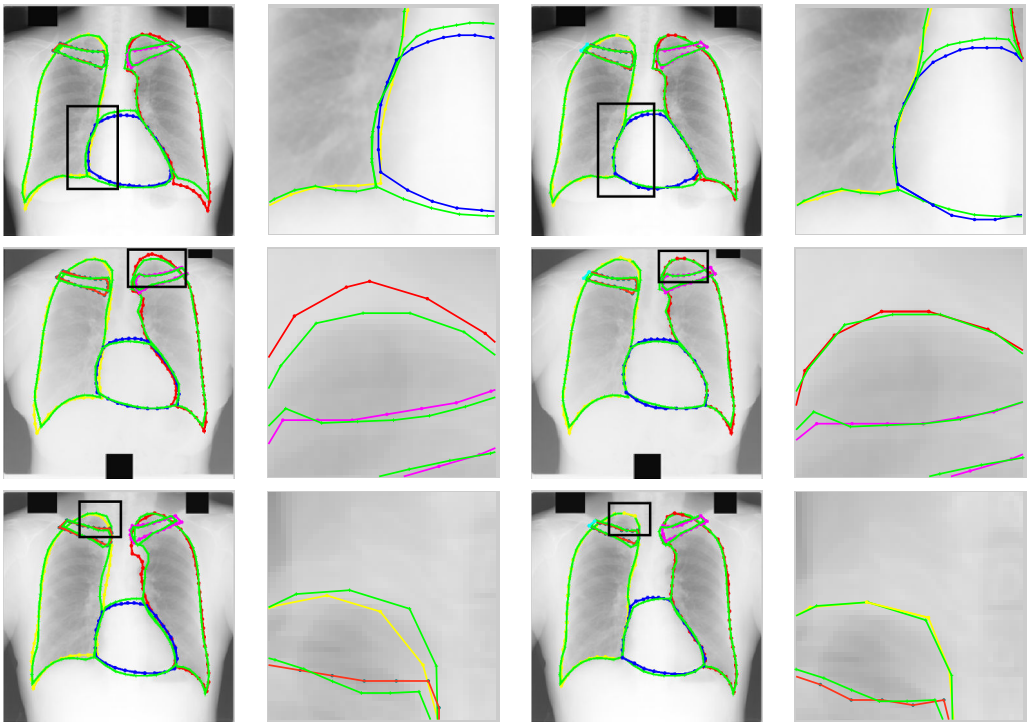
Figure 2.6: (a) SIFT computation: the image gradients are represented with arrows in blue. To compute the descriptor at a point, its corresponding grid is overlaid on top of the gradients and histograms of the gradient are computed within all of these sub-regions. The overall stacking of the histograms forms the descriptor. (b) Daisy computation [Tola *et al.* 2008]: first, gradient magnitude layers in different orientations are computed. Each of these layers are the magnitude of the gradient in a specific direction. Then, a convolution with a Gaussian kernel is applied to get the histograms for every point. These values are then concatenated to get the descriptor vector.

having built-in invariances due to multiplicative and additive signal changes; furthermore they can also take a large part of the image into account without requiring excessive blurring (unlike standard filterbanks), which has a detrimental effect in discriminative power. In our comparisons we use as a baseline the responses of multi-scale steerable filterbank, implemented along the lines of [Freeman & Adelson 1991].

We experiment with dense SIFT descriptors [Fulkerson *et al.* 2008] and Daisy descriptors. Our results indicate that our dense descriptors yield a systematic boost in performance, when compared to simpler baselines. In our comparisons we use as a baseline the responses of multi-scale steerable filterbank, implemented along the lines of [Freeman & Adelson 1991].

2.4.3.2 Two-stage Parameter Estimation versus Joint Training

We compare the performance of our DCMs in the following two settings: (a) using, as in [Seghers *et al.* 2007], parameters estimated through Maximum Likelihood estimation for the pairwise terms (means and standard deviations) and classifiers trained separately per every landmark and (b) using parameters jointly trained through our joint training objective.



a b c d

(a,b): chain graph (baseline). (c,d): loopy-graph results (ADMM results).

Figure 2.7: Segmentation results on lungs, heart and clavicles. Ground truth contours are shown in green, our results are shown in other colors. We observe that the loopy-graph model delivers more accurate results that stick more closely to the ground truth annotations. We attribute this to the ability of our loopy-graph model to account for closedness constraints, and also to model interactions among multiple parts - for instance that the clavicle boundaries need to be at a prescribed distance from the lung boundaries.

2.4.3.3 Zero-one Loss Function versus MCD Loss Function

One possible simple choice of the loss function is the 0-1 loss. For our zero-one loss this boils down to recovering any optimum of $s_{I_i}(\mathbf{X}, \mathbf{w})$ which is different from $\hat{\mathbf{X}}_i$. We compare the performance of our shape matching algorithm in the following two settings: (a) training with the more specific Mean Contour Distance loss (b) using the standard 0-1 loss.

2.4.3.4 Loop-free Graphs versus Loopy Graphs

Omitting one geometric constraint breaks up the closed chain. In that case inference becomes easier since the problem is solved directly with dynamic

Table 2.3: Performance measures for the previous state-of-the-art of [Seghers *et al.* 2007], and different choices for our method, involving Daisy features, dense SIFT at a resolution of 4, and 8 pixels per bin, the use of chain graphs (CG suffix) vs. loopy graphs (LG suffix), and the use of the MCD loss for training (MCD suffix).

	Right Clavicle (23 points)			left Clavicle (23 points)		
method	mcd	Dice	Jacc.	mcd	Dice	Jacc.
Daisy+CG	90.4	82.48	1.8	88.11	78.75	2.3
Daisy+LG	91.8	84.84	1.7	89.19	80.5	2
Daisy+LG+MCD	**93.04**	**86.99**	**1.5**	**89.95**	**81.9**	1.8
Sift-4+CG	89.9	81.65	1.9	88.1	78.73	1.8
Sift-4+LG	92.89	86.72	1.6	89.6	81.2	1.5
Sift-8+CG	90.00	81.82	1.9	87.4	77.6	2.8
Sift-8+LG	91.75	84.76	1.9	89.22	80.6	1.9
Sift-8+LG+MCD	92.8	86.57	1.7	89.8	81.5	**1.4**

programming at the expense of potentially decreased performance. We compare the performance of two trained models.(a) an open contour model and (b) a loopy model involving closed contours and intra organ connections. We note that we have at our disposal algorithms that wan deal with exact inference with these models efficiently as will be described in Chapter 3. Since both optimization and learning do not suffer from local minima issues, we can thus attribute any difference in the final results exclusively to the low-level feature, model structure, and loss function choices.

2.4.4 Quantitative evaluation of different design choices:

In Table 2.4 we report validation measures for the different design choices considered. Our very first observation is that our simplest baseline (convolution filters with two-stage learning) reaches similar performance to the state-of-the-art MISCP algorithm [Seghers *et al.* 2007], which in turn outperfomrs ASMs [van Ginneken *et al.* 2006] as extensively demonstrated in [Seghers *et al.* 2007]. We further verify that: (i) joint training boosts performance when compared to two-stage training; (ii) dense appearance descriptors have a clear edge over standard convolution features in both joint training and two-stage training; (iii) the use of loopy models improves performance; (iv) the use of the MCD loss improves performance as well.

These results are consistently supported practically by all organs, evaluation measures, and front-end feature choices. Optimizing the MCD loss during

Table 2.4: Performance measures for the previous state-of-the-art of [Seghers *et al.* 2007], and different choices for our method, involving Daisy features, dense SIFT at a resolution of 4, and 8 pixels per bin, the use of chain graphs (CG suffix) vs. loopy graphs (LG suffix), and the use of the MCD loss for training (MCD suffix).

	Right Lung (44 points)			Left Lung (50 points)			Heart (26 points)		
method	Dice	Jacc.	mcd	Dice	Jacc.	mcd	Dice	Jacc.	mcd
Daisy+CG	97.97	96.0	1.2	97.52	95.2	1.4	95	91.3	2.3
Daisy+LG	98.1	96.27	1.3	97.66	95	**1.2**	96.5	93.2	**1.3**
Daisy+LG+MCD	**98.24**	**96.54**	**1.0**	**97.89**	**95.9**	**1.7**	**96.84**	**93.9**	1.7
Sift-4+CG	97.54	95.2	1.5	96.8	93.6	1.8	94.3	91.3	2.3
Sift-4+LG	97.35	94.84	1.7	97.52	95.2	1.4	96.17	92.6	2.7
Sift-4+LG+MCD	97.88	95.85	0.9	97.8	95.7	1.9	96.95	94.5	1.8
Sift-8+CG	97.71	95.52	1.5	97.00	94.1	2.0	95	90.7	2.3
Sift-8+LG	97.68	95.47	1.3	97.28	94.6	1.5	95.81	91.1	2.8
Sift-8+LG+MCD	98.00	96.1	**0.9**	97.9	95.9	1.4	96.20	92.7	1.7

training further improves the performance of our system. This is reflected by the clear boost in performance versus the 0-1 loss training, as assessed by the MCD validation measure on the test set.

The results in Table 2.4 are complemented by the results in Table 2.2 where we provide validation measures for the heart and clavicle segmentation results and ASMs [van Ginneken *et al.* 2006] on the same dataset (results of [Seghers *et al.* 2007] on the heart and clavicle segmentation tasks were not reported in [Seghers *et al.* 2007]). Furthermore, Table 2.5 shows that our baseline (dense features and joint training) already has the lowest mean pixel error (0.022) compared to all the available methods evaluated on the SCR database. We refer to the SCR website [scr] for a detailed overview of the performance of all available methods described in [van Ginneken *et al.* 2006].

Turning to qualitative comparison, an extensive side-by-side comparison of our proposed method and the state-of-the-art MISCP algorithm [Seghers *et al.* 2007] as well as the ASM based method [van Ginneken *et al.* 2006] is provided in Fig 2.8, which demonstrates the higher accuracy attained by our model on challenging areas with poor low-level information and the substantial improvement over the state-of-the art MISCP [Seghers *et al.* 2007] and ASMs [van Ginneken *et al.* 2006]. The segmentations of [Seghers *et al.* 2007, van Ginneken *et al.* 2006] were retrieved from the SCR benchmark website [scr].

In Figure 2.5 we provide box plots of different validation measures for the different organs that we work with. Moreover, we compare in Table 2.5 our pixel error results with the current state-of-the-art results on the same dataset [van Ginneken *et al.* 2006, Seghers *et al.* 2007]. We further verify through a

Table 2.5: Pixel error results on the SCR database [Shiraishi *et al.* 2000, van Ginneken *et al.* 2006]. The proposed method scores better than the state-of-the art approaches.

method	pixel error
Our full-blown model	$\mathbf{0.017 \pm 0.008}$
Our open contour model	0.022 ± 0.006
MISCP [Seghers *et al.* 2007]	0.033 ± 0.017
ASM tuned [van Ginneken *et al.* 2006]	0.044 ± 0.014

paired T test [Goulden 2008] that the pixel error improvement is statically significant (p=0.04). We validate hence again that structured prediction with the MCD loss coupled with a loopy model results in clear, systematic improvements over the state-of-the-art for all of the organs that we consider in our evaluation.

2.5 Conclusion

In this Chapter we have introduced a discriminative method for training DCMs. We demonstrated systematic improvements over the current state-of-the-art in medical image segmentation through the use of open contour models trained with standard 0-1 loss. The use of richer models (closed contours and relative shape positions), better adapted score functions trained with structured prediction with a loss function specific to contours, as well as rich appearance features, improved further the results.

As future work, we intend to further pursue the learning of loopy graph models for other shape matching tasks, such as face recognition and body pose estimation, where matching accuracy is of importance [Andriluka *et al.* 2012, Sapp *et al.* 2010, Sapp *et al.* 2011a, Bourdev & Malik 2009]; with the advent of strongly-supervised datasets [Azizpour & Laptev 2012, Vedaldi *et al.* 2014] we anticipate that this will become increasingly central to high-level vision tasks, such as object detection.

We see our work as building a bridge between recent advances in structured prediction for pose estimation, e.g. [Sapp *et al.* 2011b] and the rich set of tools developed around shape/contour detection in the medical imaging community; in future work we intend to further pursue this research direction for tasks involving more complex energy functions as well as 3D data.

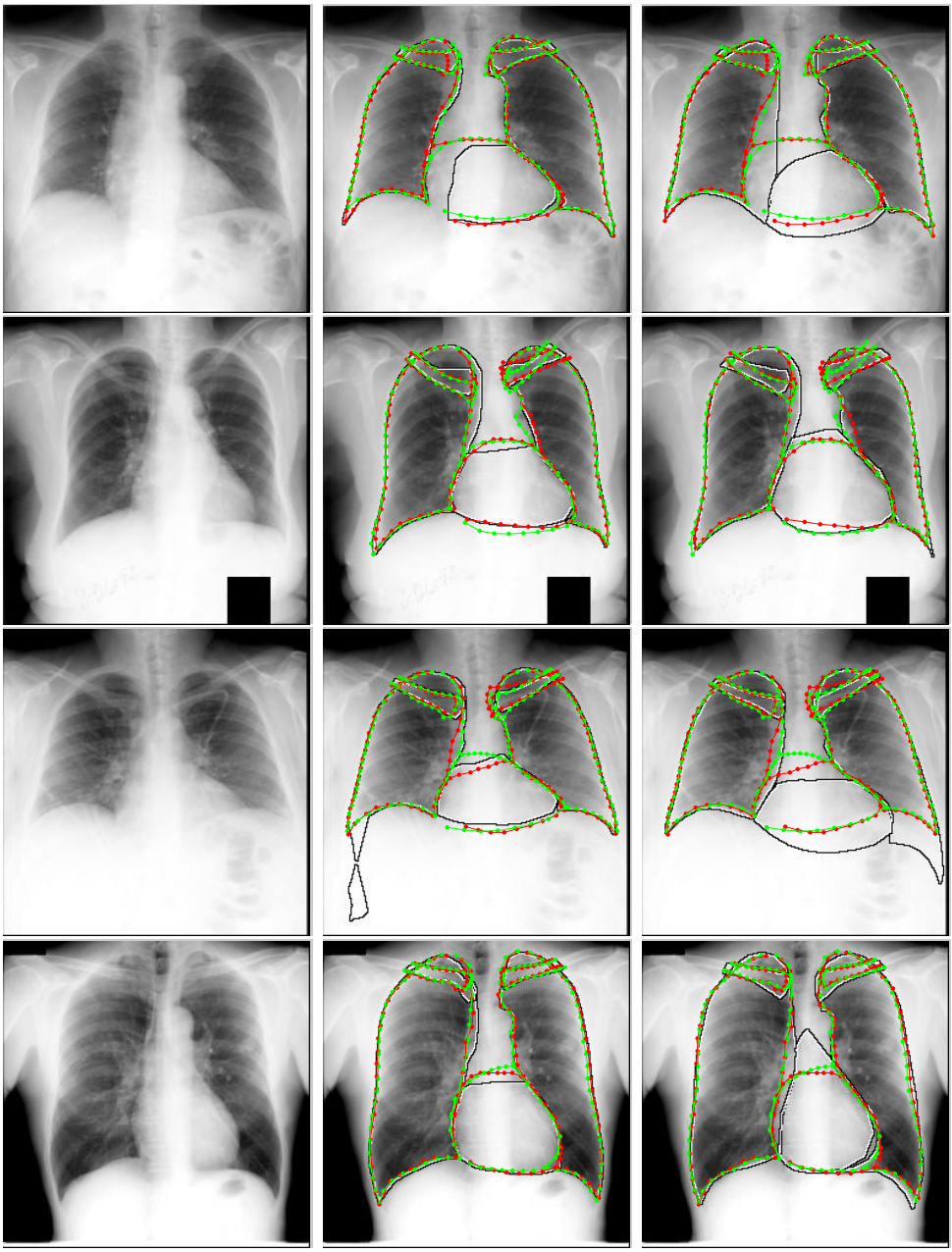


Figure 2.8: Left: patient chest radiograph. Middle: our segmentations (red) with our baseline method superimposed on the results of the MISCP algorithm [Seghers *et al.* 2007] (black contours). Right: Our segmentations (red) with our baseline method superimposed on the results of ASM based method of [van Ginneken *et al.* 2006] (black contours). The ground truth segmentations are shown in green. Each row represents the same patient radiograph.

Chapter 3

Efficient Inference for Loopy Deformable Contour Models

Contents

3.1 Introduction

In this Chapter, our goal is to segment ensembles of shapes in medical images: this involves outlining the boundaries of medical organs, while potentially exploiting inter-organ dependencies to transfer information from clearly visible parts to harder areas. We cast multi-organ shape segmentation and landmark localization in a graphical model framework, and tackle the resulting optimization problem. We use loopy graphs to incorporate problem constraints such as contour closedness and relative shape positions, that cannot be encoded through chain- or tree- structured graphs.

This directly raises the computational efficiency issue - addressing which is a main contribution of this Chapter. In our case we have many (196) nodes and a label space in the order of tens, or hundreds of thousands of values, corresponding to discretized 2D positions. As such, it is not straightforward

to apply generic techniques for approximate discrete inference on loopy graphs, as these are typically designed to deal with relatively small label sizes.

Our contribution relies in the use of an efficient decomposition-coordination algorithm to solve the resulting optimization problems. We decompose the model's graph into a set of open, chain-structured, graphs, as commonly used in Dual Decomposition techniques to optimize MRFs in computer vision applications. Earlier works [Sapp *et al.* 2011a] on applying Dual Decomposition to Deformable part Models have reported that when implemented for spatial variables Dual Decomposition is slow, or does not converge (500 iterations were used in [Sapp *et al.* 2011a]), and therefore resorted to approximate inference. Instead, in this work we use the Alternating Direction Method of Multipliers (ADMM) to fix the potential inconsistencies of the individual solutions and we show that ADMM yields substantially faster convergence than plain Dual Decomposition-based methods.

We demonstrate the merits of exact and efficient inference with rich, structured models in a large X-Ray image segmentation benchmark, where we obtain huge speedups over sub-gradient based Dual Decomposition coupled with GDTs.

3.2 Previous Work

As will be detailed in Section 3.3, one of our main technical contributions in this chapter consists in introducing ADMM to inference in loopy graphs with large label spaces, corresponding to discretized spatial variables. ADMM can be understood as a generalization of Dual Decomposition (DD) [Komodakis *et al.* 2007, Bertsekas 1999, Sontag *et al.* 2011] which in turn is already extensively used in vision and medical image segmentation [Wang *et al.* 2010b, Wang *et al.* 2011, Xiang *et al.* 2012].

ADMM has found tremendous success in image processing/compressed sensing, commonly under the name of 'Bregman iteration' methods [Goldstein & Osher 2009, Yin *et al.* 2008]. In connection with optimization problems revolving around MRFs, ADMM has recently been used in conjunction with discrete MRFs [Martins *et al.* 2011a], and used in registration in [D. Zosso & Thiran 2014] but little work has been done for MRFs with large/continuous label spaces.

Recently [Salzmann 2013] used ADMM to perform inference with polynomial energies in continuous graphical models, by iteratively linearizing a cost function used for registration; this was done to constrain the energy function to be polynomial in the unary terms. This is however not an option for our case, where we want to match deformable shapes to unconstrained images -

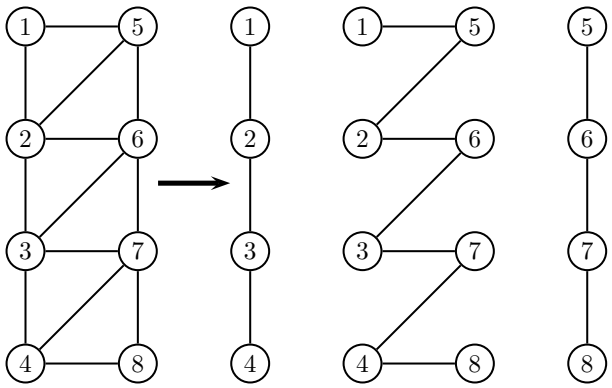


Figure 3.1: We decompose energy functions on loopy graphs into functions on chain-structured subgraphs, and use the latter as slaves in a decomposition-coordination optimization algorithm. Shown in (a) is an example of a complex graph involving a 'zipper' chain between shapes (e.g. nodes 1-4 can belong to the lung, and nodes 5-8 to the heart) and in (b) the decomposition of the complex graph into chain structured subproblems.

where the unary terms are far from linear, or convex.

3.3 Problem Decomposition

Our goal is, given an image I to solve:

$$\mathbf{X}_I^* = \underset{\mathbf{X}}{\operatorname{argmax}}\, S_I(\mathbf{X}) \tag{3.1}$$

$$\mathbf{X}_I^* = \underset{\mathbf{X}}{\operatorname{argmax}} \sum_{i=1}^{K} \mathcal{U}_{I,i}(\mathbf{x}_i) + \sum_{i,j \in \mathcal{E}} \mathcal{P}_{i,j}(\mathbf{x}_i, \mathbf{x}_j), \tag{3.2}$$

where $S_I(\mathbf{X})$ is the scoring function described in Chapter 1 and its parameters were learned in Chapter 2.

We make no assumption about the model structure as shwon in Figure 3.1, and as such contains loops; this directly reflect the problem structure (e.g. the closedness constraint of a region's boundary) and delivers more accurate segmentation results as was shown in Chapter 4, but when working with spatial variables in a large label space it is practically prohibitive to work even with the easiest loopy graphs; for instance, in the graph shown in Figure 3.1, the complexity of MAP inference grows by $O(N^3)$ where N is the number of pixels, using the junction tree algorithm [Aji & McEliece 2000].

Instead we can use problem decomposition as illustrated in Figure 3.1. In particular, we rewrite the score S_I of our graphical model as a sum of score functions $S_{I,i}$ defined on overlapping subgraphs (slaves). Formally this

is written as:

$$S_I(\mathbf{X}) = \sum_{i=1}^{N} S_{I,i}(\mathbf{X}). \tag{3.3}$$

This allows (temporarily) each slave $S_{I,i}(\mathbf{X})$ to have its own solution, $\mathbf{X}_i$, subject to the constraint that, on common nodes, different slaves must have identical solutions. As illustrated in Figure 3.1, in our problem we break every closed contour into two open chains that overlap at their end and start nodes, and introduce 'zipper' chains among organs that share edges, where the 'zipper' passes through the intra-organ edges. These are the slave problems S_i of our problem. Denoting by $R \subset 1..K$ the subset of point indices belonging to more than one chain model, our inference problem becomes:

$$\begin{aligned} \max\ S(\mathbf{X}) &= \sum_{i=1}^{N} S_i(\mathbf{X}_i) \\ \text{s.t.} \mathbf{X}_i(r) &= \mu(r),\ \forall r \in R, \quad \forall i, \end{aligned} \tag{3.4}$$

where $\mathbf{X} = \{\mathbf{X}_i\}, i = 1 \dots N$ is the ensemble of slave solutions. These slave are sharing common nodes $\mathbf{X}_r, r \in R$. r is the global index of the overlapping points. Since each slave $\mathbf{X}_i$ is allowed to have its own solutions $\mathbf{X}_i(r)$ at nodes r, we introduce μ as a reference vector for those nodes and use it to ensures consistency at the overlapping points.

3.4 Dual Decomposition Inference

Dual Decomposition relaxes the constraints in Equation 3.5 by introducing a Lagrange multiplier $\lambda_i(r)$ for each agreement constraint. The Lagrangian for problem Equation 3.5 is expressed as:

$$\mathcal{A}(\{\mathbf{X}_{i=1..N}\}, \mu, \lambda) = \sum_{i=1}^{N} S_i(\mathbf{X}_i) + \sum_{r \in R} \sum_{i=1}^{\eta(r)} \lambda_i(r) \left(\mathbf{X}_i(r) - \mu(r)\right) \tag{3.5}$$

where $\eta(r)$ is the subset of slaves sharing the same node $\mathbf{X}(r)$ μ and μ is a reference vector. We note that this is deviating from the standard presentation of Dual Decomposition [Bertsekas 1999] where we have explicitly the constraint of $(\mathbf{X}_i(r) - \mathbf{X}_i(r))$, but it makes the notation convenient and the transition to ADMM technique easier.

The dual function can be written as:

$$\mathcal{L}(\lambda) = \max_{\mathbf{X}_i, \mu} \mathcal{A}(\mathbf{X}_i, \mu, \lambda) \tag{3.6}$$

$$= \max_{\mathbf{X}_i, \mu} \sum_{i=1}^{N} \left(S_i(\mathbf{X}_i) + \sum_{r \in R} \lambda_i(r) \mathbf{X}_i(r) \right) - \sum_{r \in R} \sum_{i=1}^{N} \lambda_i(r) \mu(r) \tag{3.7}$$

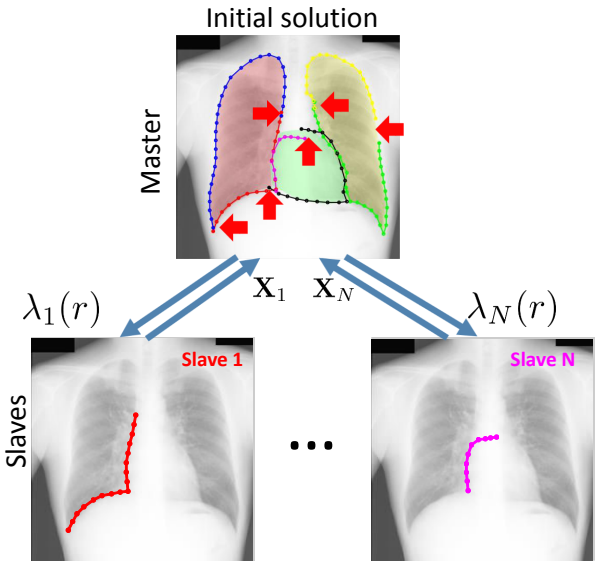


Figure 3.2: Dual Decomposition illustration: at each iteration, every slave i communicates its solution $\mathbf{X}_i$ to the master; the master then detects inconsistencies in the individual slave solutions (indicated by red arrows) and drives the slaves towards a consistent solution in the next iteration, by passing parameters $\lambda_i(r)$ that affect the slave problems around the common nodes, r.

Since the μ vector is unconstrained, maximizing over μ results in the following expression for λ:

$$\mathcal{L}(\lambda) = \begin{cases} \max_{\mathbf{X}_i} \sum_{i=1}^{N} \left(S_i(\mathbf{X}_i) + \sum_{r \in R} \lambda_i(r)\mathbf{X}_i(r)\right) & \text{if } \sum_{i=1}^{N} \lambda_i(r) = 0, \quad \forall r \in R \\ +\infty & \text{otherwise.} \end{cases}$$

We call Λ the feasible set and express it as $\Lambda = \{\lambda, \sum_{i=1}^{N} \lambda_i(r) = 0, \forall r \in R\}$. Since positions where $\mathcal{L}(\lambda) = +\infty$ is no use to us, we consider instead the following optimization problem using Λ:

$$\mathcal{L}(\lambda) = \max_{\mathbf{X}_i} \sum_{i=1}^{N} \left(S_i(\mathbf{X}_i) + \sum_{r \in R} \lambda_i(r)\mathbf{X}_i(r)\right), \text{s.t} \lambda \in \Lambda \tag{3.8}$$

We underline here that the dual function decomposes in the slave variables $\mathbf{X}_i$ as:

$$\mathcal{L}(\lambda) = \sum_{i=1}^{N} \max_{\mathbf{X}_i} \left(S_i(\mathbf{X}_i) + \sum_{r \in R} \lambda_i(r)\mathbf{X}_i(r)\right) \tag{3.9}$$

$$= \sum_{i=1}^{N} \mathcal{L}_i(\lambda_i) \tag{3.10}$$

The term Dual Decomposition stems from the decomposition of the dual function. Evaluating the dual function is hence equivalent to solving two independent optimization problems. For every λ, the value of the dual function $\mathcal{L}(\lambda)$ is an upper bound on the solution to the primal problem. The dual problem is then to compute the optimal bound and find λ^* such that:

$$\begin{aligned} \lambda^* &= \operatorname*{argmin}_{\lambda} \mathcal{L}(\lambda) \\ \text{s.t} &\quad \lambda \in \Lambda \end{aligned} \tag{3.11}$$

This dual function $\mathcal{L}$ is convex in λ but is not differentiable; this motivates the use of the projected sub-gradient descent method described in [Bertsekas 1999]. This implies an iterative process, where at each iteration the current value of each dual variable $\lambda_i(r)$ is updated according to the sub-gradient until convergence as follows:

$$\lambda_i^{t+1}(r) = \lambda_i^t(r) - \alpha_t [\nabla \mathcal{L}_i(\lambda_i^{t+1}(r))]_\Lambda \tag{3.12}$$

where α_t is the step size. A sub-gradient can be obtained by:

$$\nabla \mathcal{L}_i(\lambda_i^{t+1}(r)) = \mathbf{X}_i^{t+1}(r) \tag{3.13}$$

where

$$\mathbf{X}_i^{t+1}(r) = \operatorname*{argmax}_{\mathbf{X}_i(r)} \mathcal{L}_i(\lambda_i^t) \tag{3.14}$$

$$= \operatorname*{argmax}_{\mathbf{X}_i(r)} \left(S_i(\mathbf{X}_i) + \sum_{r \in R} \lambda_i(r) \mathbf{X}_i(r) \right). \tag{3.15}$$

We see that solving Equation 3.15 for a given slave i amounts to performing inference independently in a chain structured model. We can verify that the effect of the Lagrangian function on the individual subproblems is absorbed by updating the unary terms of the slaves with a function of position.

Projecting the sub-gradient to the feasible set amounts to a centering operation expressed as:

$$[\nabla \mathcal{L}_i(\lambda_i^{t+1}(r))]_\Lambda = \mathbf{X}_i^{t+1}(r) - u^{t+1}(r) \tag{3.16}$$

where $u^{t+1}(r)$ is the average value of individual solutions:

$$u^{t+1}(r) = \frac{\sum_i \mathbf{X}_i^{t+1}(r)}{\eta(r)} \tag{3.17}$$

Putting all the pieces together, we can derive now a Dual Decomposition algorithm. At each iteration, the master problem 3.12 updates the dual variables

in the direction of the sub-gradient of the dual function. The sub-gradient at the current dual variable is provided by the slave problems 3.15. This is schematically described in Figure 3.2.

Upon convergence, we have access to the tightest upper bound $\mathcal{L}(\lambda^\star)$ of the optimal score, and also to solutions $\mathbf{X}^\star_{i=1..N}$ of the original problem. A lower bound on the optimal score is then given by $\sum_{i=1}^{N} S_i(\mathbf{X}_i^\star)$. We can compute the duality gap which gives an indication of how close we are to the optimal score.

3.4.1 Dual Decomposition Variants

In order to guarantee convergence of Dual Decomposition, we need to set $\alpha_t, t = 1 \dots T$ as a diminishing and non-summable sequence i.e:

$$\alpha_t \geq 0 \, \forall t; \lim_{t \to \infty} \alpha_t = 0; \sum_{t=1}^{\infty} \alpha_t = \infty \tag{3.18}$$

as demonstrated by [Bertsekas 1999]. Different variable update strategies were introduced in the literature to decrease the number of alterations and described in [Komodakis *et al.* 2011]. We experimented with the sequence $\alpha_t = \frac{a}{\sqrt{t}}$. A more sophisticated sequence is introduced in [Duan *et al.* 2012] based on Polyak's step size rule [Polyak 1967]. The dual variable update strategy required by the subgradient descent technique implies that DD ends up taking increasingly small steps toward the optimum, and hence exhibiting a slow convergence rate. In the following we describe ADMM, a decomposition-coordination algorithm that converges faster than DD.

3.5 ADMM Inference

We address the problem in Equation 4.8 by building on the Dual Decomposition (DD) technique [Bertsekas 1999, Komodakis *et al.* 2007], and in particular its acceleration attained with the Alternating Direction Method of Multipliers [Boyd *et al.* 2011, Martins *et al.* 2011b]. This combines the benefits of DD [Bertsekas 1999] (fast optimization of the slave problems) and ADMM (rapid convergence) practically without altering the optimization procedure for the subproblems.

ADMM adds a quadratic penalty for constraint violation, yielding the *augmented Lagrangian*:

$$\mathcal{A}_\rho(\mathbf{X}_i, \mu, \lambda) = \sum_{i=1}^{N} S_i(\mathbf{X}_i) + \sum_{r \in R} \sum_{i=1}^{N} \lambda_i(r) \left(\mathbf{X}_i(r) - \mu(r)\right) - \rho \sum_{r \in R} \sum_{i=1}^{N} (\mathbf{X}_i(r) - \mu(r))^2$$

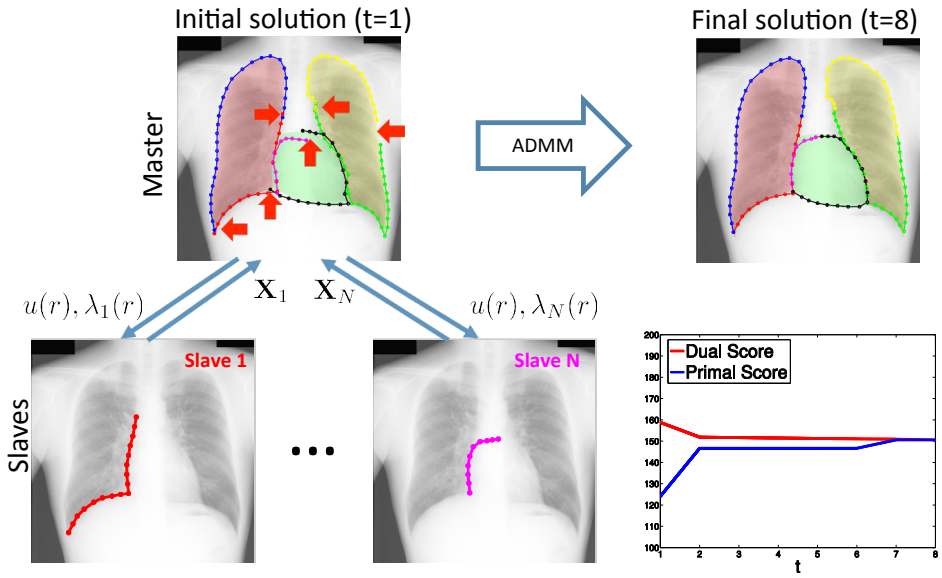


Figure 3.3: At each iteration, every slave i communicates its solution $\mathbf{X}_i$ to the master; the master then detects inconsistencies in the individual slave solutions (indicated by red arrows) and drives the slaves towards a consistent solution in the next iteration, by passing parameters $u(r), \lambda_i(r)$ that affect the slave problems around the common nodes, r. ADMM quickly leads to consensus among the different slaves, as shown on the right: the dual and the primal problems reach a zero duality gap in a small number of iterations. The estimated organ boundaries closely match the color-coded ground truth organ segmentation.

where we augment the Lagrangian in Equation 4.33 with an euclidean penalty on the discrepancy between the individual solutions and the reference vector μ and introduce ρ as a positive parameter that controls the intensity of the augmenting penalty. The aim of this extra quadratic penalty term is to improve the convergence properties of the corresponding minimization algorithm. We note that we deviate a bit from the common presentation of the method, e.g. [Boyd *et al.* 2011], as we phrase our original problem as one of maximization rather than minimization.

We can see this augmented Lagrangian as formulating the Lagrangian of the following problem:

$$\max\ S(\mathbf{X}) = \sum_{i=1}^{N} S_i(\mathbf{X}_i) - \rho \sum_{r \in R} \sum_{i=1}^{N} (\mathbf{X}_i(r) - \mu(r))^2 \tag{3.19}$$

$$\text{s.t. } \mathbf{X}_i(r) = \mu(r)\ \forall r \in R, \quad \forall i \tag{3.20}$$

This is an equivalent problem to the one expressed by Equation 3.5 as the

added quadratic term vanishes at convergence. However, with this formulation we do not only enforce the agreement constraint between the slave but we encourage it through the quadratic penalty. Now we write the corresponding dual problem:

$$\begin{aligned}\mathcal{L}_\rho(\lambda) &= \max_{\mathbf{X}_i,\mu} \mathcal{A}_\rho(\mathbf{X}_i,\lambda,\mu) \\ &= \max_{\mathbf{X}_i,\mu} \sum_{i=1}^{N}\left(S_i(\mathbf{X}_i)+\sum_{r\in R}\left(\lambda_i(r)\mathbf{X}_i(r)-\rho\mathbf{X}_i(r)^2+2\rho\mathbf{X}_i(r)\mu(r)\right)\right) \\ &- \sum_{r\in R}\sum_{i=1}^{N}\lambda_i(r)\mu(r)\end{aligned}$$

In order to minimize the dual function, we can apply a gradient descent method as follows:

$$\{\mathbf{X}_i^{t+1},\mu^{t+1}\} = \operatorname*{argmax}_{\mathbf{X}_i,\mu}\mathcal{A}(\mathbf{X}_i,u^t,\lambda^t) \tag{3.21}$$

$$\lambda_i^{t+1}(r) = \lambda_i^t(r)-\alpha\rho(\mathbf{X}_i^{t+1}(r)-\mu^{t+1}(r)) \tag{3.22}$$

where α is a step size. This is known as the *Method of Multipliers*.

However, due to the added quadratic penalty the dual problem is not anymore decomposable in the slave variables $\mathbf{X}_i$ as in Equation 3.9 and we need to perform a joint maximization over $\mathbf{X}_i$ and μ in Equation 3.21. As such, we lose the decomposition property of DD, unless $\rho = 0$ which brings us back to the original DD scheme. Instead, ADMM iterates the following steps:

$$\mathbf{X}_i^{t+1} = \operatorname*{argmax}_{\mathbf{X}_i}\mathcal{A}(\mathbf{X}_i,\mu^t,\lambda^t) \tag{3.23}$$

$$\mu^{t+1} = \operatorname*{argmax}_{u}\mathcal{A}(\{\mathbf{X}_i^{t+1}\},\mu,\lambda^t) \tag{3.24}$$

$$\lambda_i^{t+1}(r) = \lambda_i^t(r)-\alpha\rho(\mathbf{X}_i^{t+1}(r)-\mu^{t+1}(r)) \tag{3.25}$$

where we decouple the joint optimization in Equation 3.21 into as sequential optimization of $\mathbf{X}_i^{t+1}$ and u^{t+1}. The term *Alternating Direction* stems for the update of $\mathbf{X}_i^{t+1}$ and u^{t+1} in an alternating fashion.

In words, the slaves efficiently solve their sub-problems (Equation 3.23), and deliver $\mathbf{X}_i$ to the master. The master then coordinates the individual solutions, by updating the current multipliers $\lambda_i^{t+1}(r)$ (Equation 3.25) and $u^{t+1}(r)$ (Equation 3.24), and communicating them to the slaves for the next iteration. In [Martins *et al.* 2011b], Equation 3.24 is shown to turn out to be equal to the averaged vote for the position of the point if the penalty ρ is kept constant. Regarding the slave problems, we write Equation 3.23 as:

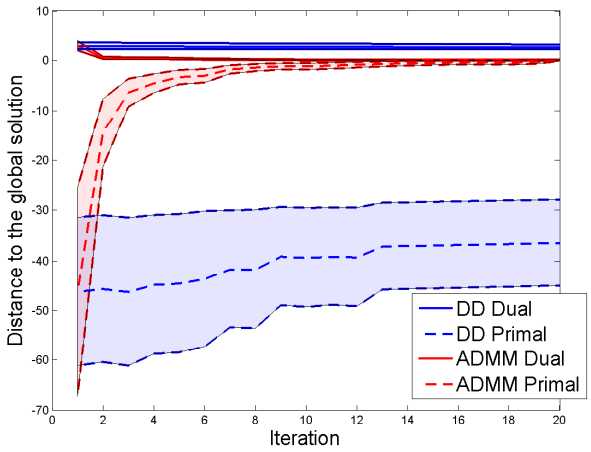


Figure 3.4: Evolution of the dual objective and the primal one as a function of DD/ADMM iterations. ADMM-based optimization rapidly converges, achieving a duality gap of zero typically in less than 20 iterations. Sub-gradient based method does not converge even after 100 iterations. These results are obtained by averaging over hundred different example images.

$$\mathbf{X}_i^{t+1} = \underset{\mathbf{X}_i}{\operatorname{argmax}}\, \mathcal{A}(\mathbf{X}_i, u^t, \lambda^t) \tag{3.26}$$

$$= \underset{\mathbf{X}_i}{\operatorname{argmax}} \sum_{i=1}^{N} S_i(\mathbf{X}_i) + \sum_{r\in R}\sum_{i=1}^{N} \lambda_i^t(r)\left(\mathbf{X}_i(r) - u^t(r)\right)$$

$$- \rho \sum_{r\in R}\sum_{i=1}^{N} (\mathbf{X}_i(r) - u^t(r))^2 \tag{3.27}$$

Since the maximization runs only over $\mathbf{X}_i$ and λ and μ are now fixed, we can then decompose the optimization of this problem in the slaves as follows:

$$\mathbf{X}_i^{t+1} = \underset{\mathbf{X}_i}{\operatorname{argmax}}\, S_i(\mathbf{X}_i) + \sum_{r\in R} \lambda_i^t(r)\left(\mathbf{X}_i(r) - u^t(r)\right) - \rho \sum_{r\in R} (\mathbf{X}_i(r) - u^t(r))^2$$

$$= \underset{\mathbf{X}_i}{\operatorname{argmax}}\, \hat{S}_i(\mathbf{X}_i) \tag{3.28}$$

We can run the optimization of each slave in parallel. We note here that now the master broadcasts to the slaves the average solution μ^t. The slaves regularize their problems so as to converge to this consensus value. The master then gathers the individual solutions and updates the average value and the dual variables.

3.5.1 Application to Shape Segmentation with Loopy Graphs

We observe that solving Equation 3.23 for a given slave i amounts to performing inference independently in a chain structured model. We can verify that the effect of the (augmented) Lagrangian function on the individual subproblems is absorbed by updating the unary terms of the slaves with a parametric, quadratic function of position; since the slaves are chain-structured, this means that we can still efficiently optimize them with GDTs. But we will see in Section 4.5, how we can adapt the inference algorithm in Chapter 4 to accelerate the optimization.

After optimizing all slaves, we can recover the dual score by summing over the optimal values of the considered functions and write:

$$\mathcal{L}(\lambda) = \sum_{i=1}^{N} \hat{S}_i(\mathbf{X}_i^*) \tag{3.29}$$

We can recover the primal score by recalling that our merit function is an inner product that writes:

$$\mathcal{P}(\mathbf{X}_i) = \sum_{i=1}^{N} \langle \mathbf{w}, \mathbf{h}_I(\mathbf{X}_i^*) \rangle \tag{3.30}$$

The primal and dual function evaluation at each iteration allows us to compute the gap between the dual score and the primal score and assess therefore the convergence of ADMM and DD. As shown in Figure 3.4, ADMM is dramatically faster than Dual Decomposition for our problem. For our full-blown model, involving $|R| = 30$ *shared* nodes in Equation 3.20, Dual Decomposition would often not converge even after 100 iterations, while we obtained convergence of ADMM in typically no more than 20 iterations. This means that the effective complexity of our joint inference algorithm in loopy graphs bowls down to be roughly equal to the complexity of the optimization of the slaves.

We note that ADMM is guaranteed to converge to the global optimum only if the score function being maximized is concave. In our case, this is not guaranteed (the unary terms are arbitrary), so we can understand ADMM only as an approximate optimization algorithm. The fact that we have a zero duality gap at convergence (as shown in Figure 3.4) indicates that for the examples considered in our experiments approximate inference delivers the exact solution.

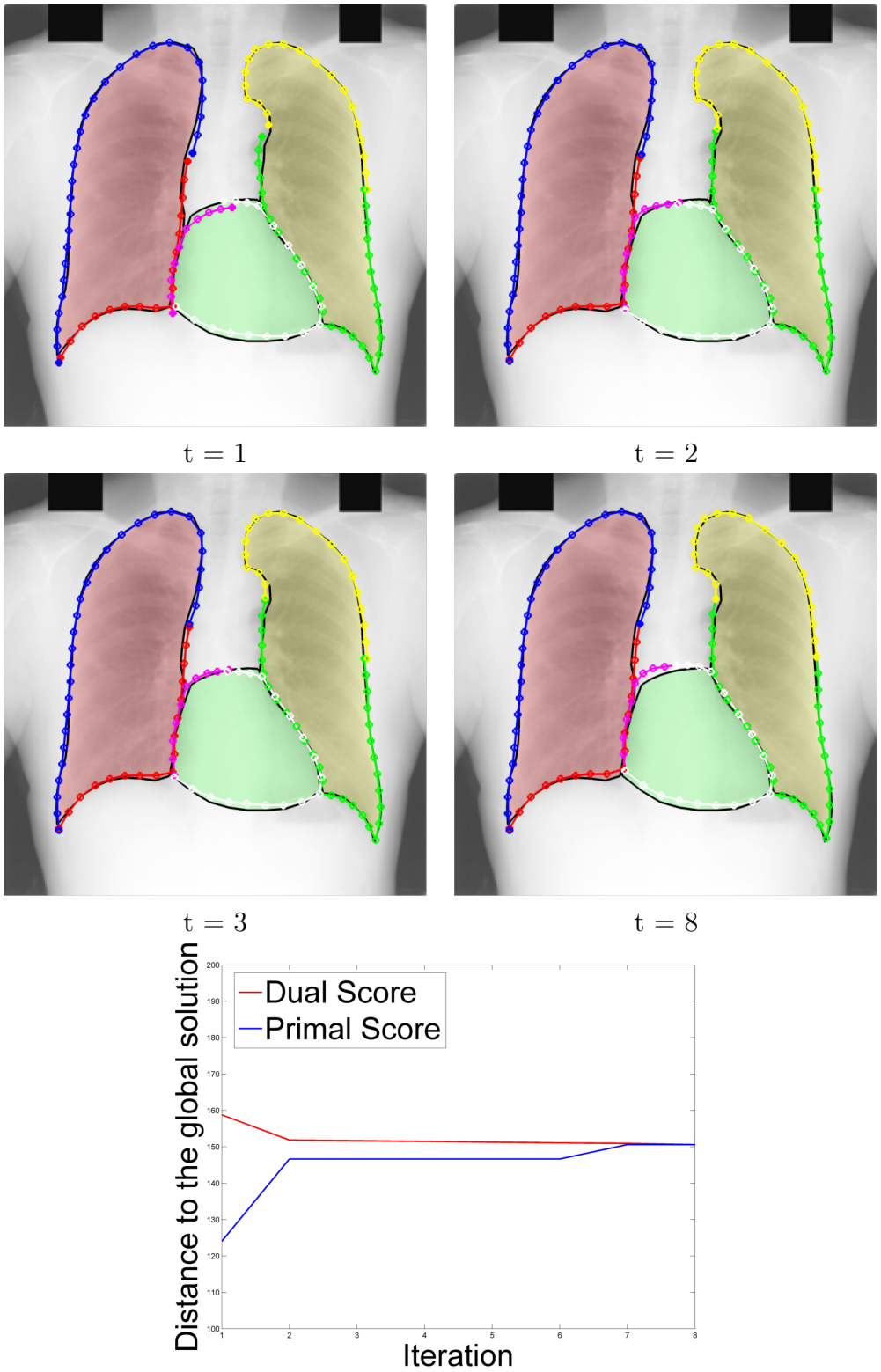


Figure 3.5: Iterations of ADMM inference on an image from the considered database; in the first image the segmentation exhibit inconsistencies in the individual slave solutions. As shown in the 8th iteration, ADMM quickly leads to consensus among the different slaves leading the dual and the primal problems to reach a zero duality gap.

3.5.1.1 ADMM hyperparameters

We have to tune two hyperparameters in ADMM consisting in the penalty coefficient ρ and the stepsize coefficient α. We have set ρ and α empirically. In practice, we observed that ρ has an important impact on the performance of ADMM. On the one hand, setting ρ to small values results in a slower convergence of the algorithm. This is anticipated since setting $\rho = 0$ brings us back basically to a DD implementation. On the other hand, we observed that when setting ρ to big values (> 10), we have oscillations in the dual function, typically ($3\,or\,4$) but at the cost of a convergence to a local optimum manifested by a non-zero duality gap. Picking $1 \leq \rho \leq 4$ gave us the best performance of ADMM in practice.

Regarding the step size coefficient α, we observed that it plays a less important role in ADMM performance that the penalty coefficient. With appropriate values it can indeed decrease relatively the number of iterations but does not have an impact on the duality gap at convergence. Other implementations of ADMM [Martins *et al.* 2011b] get rid of this parameters by setting $\alpha = 1$ so that we have the penalty coefficient equal to the stepsize in the dual variable update in Equation 3.25

3.6 Multi-Scale Optimization

We note that we accommodate global scale changes through multi-resolution optimization; from our original image I we construct an image pyramid by resampling at a set of scales $\mathcal{S} = (1, r, r^2, \ldots, r^S)$ and compute:

$$\mathbf{X}^*_{I,\mathcal{S}} = \underset{\mathbf{X},i}{\operatorname{argmax}}\, s_{I(r^i)}(\mathbf{X}, \mathbf{w}), \tag{3.31}$$

where $I(r^i)$ denotes the image resampled with a ratio r^i.

3.7 Results

Our model contains 196 nodes including 30 shared nodes. We have 16 slave problems, 2 per organ plus 26 for the links ('zipper' contours). For ADMM we found that we achieve fastest convergence when setting $\rho = 1$ in 4.33 and setting the step size α_t to follow a non-summable diminishing step length rule, as detailed in [Bertsekas 1999].

Starting with computational efficiency, for our problem the computation of unary terms requires 0.4 seconds on a standard PC; each slave problem takes 0.06 seconds per contour and scale, for a contour with 20 nodes; for 8

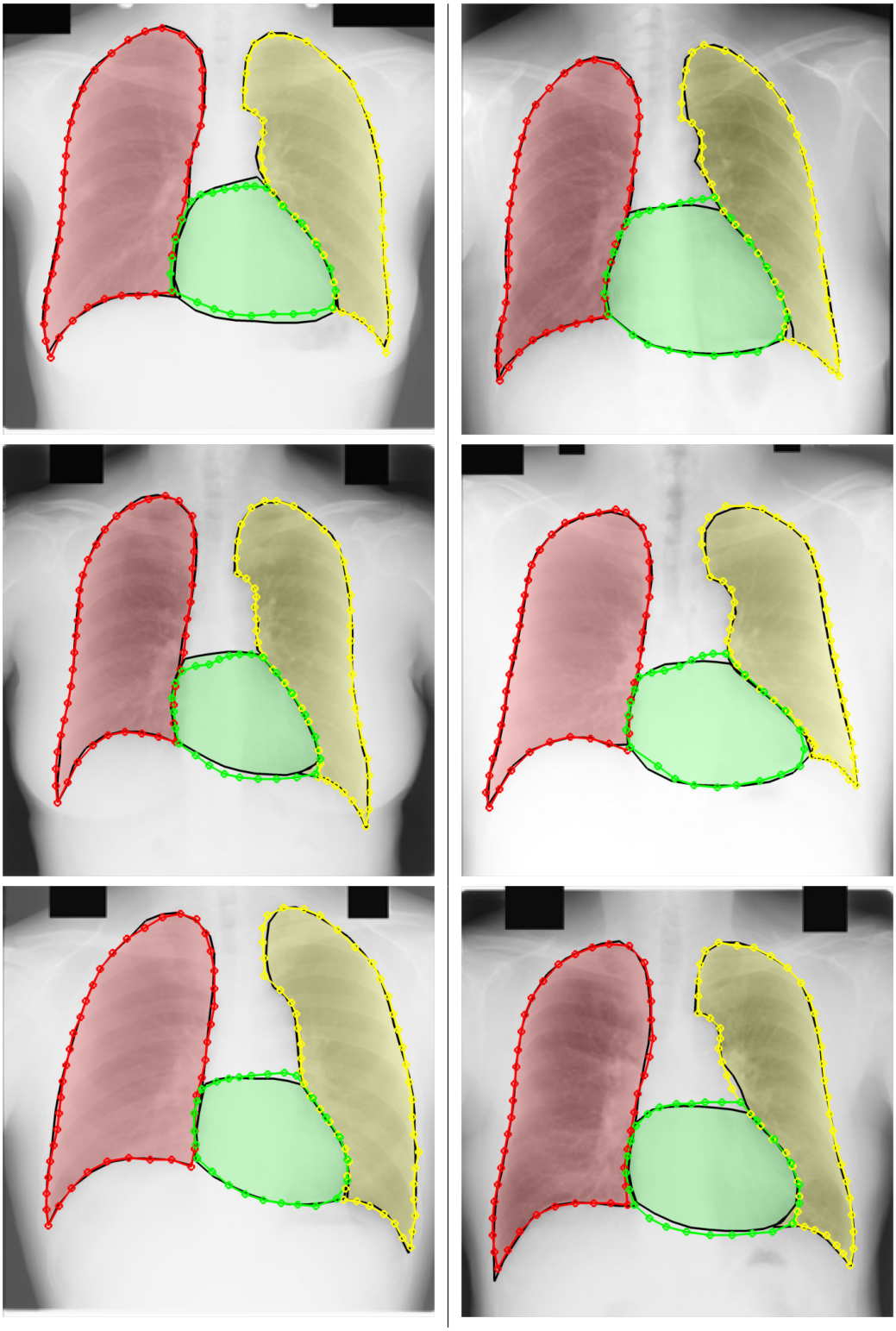


Figure 3.6: Qualitative results produced by our loopy model.

contours and 7 scales, this means 3.4 seconds per iteration of ADMM/Dual Decomposition. ADMM/DD take practically the same time per iteration, but ADMM converges in less than 20 iterations (typically 10) while DD did not converge even after 100 iterations.

A side-by-side comparison of our baseline model from Chapter 4 involving open contours and the full-blown, loopy-graph model developed in this Chapter is provided in Figure 2.7, which qualitatively demonstrates the higher accuracy attained by our model on challenging areas with poor low-level information. Some notable cases include the case of the heart, where the added geometric constraint allows us to recover from unary detector failure in the blank area.

3.8 Conclusion

In this chapter, we developed an efficient technique to performing inference on loopy graphs with spatial variables, by employing the Alternating Directions Method of Multipliers (ADMM). We applied our algorithm to segment multiple organs from medical images and achieved a speedup of orders of magnitude over plain DD optimization coupled with GDTs.

Apart from the obvious societal impact of medical imaging, what we find most interesting in this problem is the complexity, and accuracy of the annotation being employed -we have 196 landmarks, localized by expert physicians. Such datasets provide a challenging testbed for algorithm assessment, while with the advent of strongly supervised object annotations [Azizpour & Laptev 2012, Bourdev & Malik 2009, Vedaldi *et al.* 2014] we anticipate that our advances will become increasingly relevant to recognition.

While ADMM is only one of many options for accelerating over DD, we showed that its adaptation to inference problems with spatial variables is particularly simple and yet powerful. We anticipate hence that our work could serve as an efficient and reliable computational module for larger inference problems such as object detection and pose estimation.

Chapter 4

Efficient Inference for Chain-structured Deformable Contour Models

4.1 Introduction

In this chapter, we focus on accelerating inference on our ADMM subproblems described in Chapter 3. We consider a chain-structured Deformable Contour Model (DCM) for anatomical structure shape segmentation. We use a simpler connectivity than the arbitrary connectivity in the model introduced in Chapter 1 Section 1.2.2; we define an ordering constraint between landmarks to represent a single continuous contour.

Our contributions are twofold: firstly, is the derivation of a rapid inference algorithm for chain structured graphs. We capitalize on recent advances in fast Deformable Part Models (DPMs) optimization [Kokkinos 2011b] via branch-and-bound and include them in the Hierarchical A^* Lightest Derivation (HA^*LD) [Felzenszwalb & Mcallester 2007] architecture. Our second contribution consists in solving the slave problems in our ADMM scheme described in Chapter 3 efficiently.

Performing inference efficiently on this chain-structured graph is a main challenge, since this involves a large label space consisting of discretized 2D positions. In order to achieve computational efficiency, current works rely on Dynamic Programming (DP) [Felzenszwalb & Zabih 2011, Geiger *et al.* 1995] to recover the globally optimal solution in a time quadratic in the number of pixels. Constraining the model furthermore to use separable quadratic pairwise terms allows us to couple DP with the Generalized Distance Transform (GDTs) technique [Felzenszwalb & Huttenlocher 2004, Felzenszwalb & Huttenlocher 2005]. This reduces the complexity to be linear in the number of pixels.

Recent approaches improve inference efficiency in star-shaped DPMs. [Sapp *et al.* 2010, Ferrari *et al.* 2008, Kumar & Torr 2006] simplify temporarily the cost function being optimized in order to reduce the high-dimensional search space. Pruned dynamic programming is used in [Chen *et al.* 2007]

to efficiently detect DPMs, while coarse-to-fine dynamic programming [Raphael 2001] aims at accelerating the plain dynamic programming method. [Felzenszwalb *et al.* 2010] use precomputed thresholds so as to prune computation with minimal possible loss. Finally, [Kokkinos 2011b] propose an exact Branch-and-Bound technique to solve star-shaped graph optimization in a time practically logarithmic in the number of pixels.

This Chapter is organized a follows: we firstly describe the current baseline method for optimizing chain-structured DPMs. This technique is based on DP coupled with GDTs. We then devise a coarse-to-fine algorithm following the architecture of the Hierarchical A^* Lightest Derivation (HA^*LD) [Felzenszwalb & Mcallester 2007] scheme while including the pruning technique developed in [Kokkinos 2011b]. Since multi-scale optimization is required to deal with scale changes in our image, we show that we can obtain further acceleration in multi-scale optimization scheme by using a single priority queue for all scales. We then integrate the proposed inference algorithm in the ADMM scheme presented in Chapter 3 to ensure an efficient optimization of the slave problems.

We apply our method to the problem of shape segmentation of several anatomical structures in Posterior-Anterior chest radiographs. We verify that our proposed algorithm delivers exactly the same results, but with a 10-fold speedup on average over the state-of-the-art optimization based on multi-scale DP accelerated with GDTs.

4.2 Merit Function

Ou merit function is an instance of the generic merit function introduced in Chapter 1 with Equation 1.9, in the sense that we consider a particular connectivity between landmarks. Namely, we represent a 2D shape as an open contour consisting of a sequence of K anatomical landmarks: $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_K\}$. We recall that every landmark is a 2D position vector $\mathbf{x}_i = (h_i, v_i)$. Given an image I we score a contour $\mathbf{X}$ with a merit function S formed as

$$S_I(\mathbf{X}) = \sum_{i=1}^{K} \mathcal{U}_{I,i}(\mathbf{x}_i) + \sum_{i=1}^{K-1} \mathcal{P}_i(\mathbf{x}_i, \mathbf{x}_{i+1}). \tag{4.1}$$

The unary terms keep the same form as in Chapter 1 with Equation 1.10. The pairwise terms also keep the same form as in Chapter 1 with Equation 4.2, except that now they constrain the location $\mathbf{x}_{i+1} = (h_{i+1}, v_{i+1})$ of landmark $i+1$ with respect to its antecedent landmark $\mathbf{x}_i = (h_i, v_i)$ in terms of a

quadratic expression of the form:

$$\begin{aligned}\mathcal{P}_i(\mathbf{x}_i, \mathbf{x}_{i+1}) &= -\left(\mathbf{x}_{i+1} - \mathbf{x}_i - \mu_i\right)^T C_i \left(\mathbf{x}_{i+1} - \mathbf{x}_i - \mu_i\right) && (4.2)\\ &= -H(h_{i+1} - h_i - \bar{h})^2 - V(v_{i+1} - v_i - \bar{v})^2. && (4.3)\end{aligned}$$

where $C_i = \text{diag}(H, V)$ is a a diagonal precision matrix and $\mu_i = (\bar{h}, \bar{v})^T$ is the nominal displacement between $\mathbf{x}_i$ and $\mathbf{x}_{i+1}$. In graphical model terminology our model is a chain-structured graph, with graph nodes corresponding to landmark locations and the edges connecting consecutive landmarks.

4.3 Previous Work

In this section we review previous work on inference approaches for chain-structured graphs, and in particular the model expressed by Equation 4.1. Formally, the task is to find the optimal configuration $\mathbf{X}^*$ given an image I such that:

$$\begin{aligned}\mathbf{X}_I^* &= \underset{\mathbf{X}}{\text{argmax}}\, S_I(\mathbf{X}, \mathbf{w}). && (4.4)\\ &= \underset{\mathbf{X}}{\text{argmax}} \sum_{i=1}^{K} \mathcal{U}_{I,i}(\mathbf{x}_i) + \sum_{i=1}^{K-1} \mathcal{P}_i(\mathbf{x}_i, \mathbf{x}_{i+1}). && (4.5)\end{aligned}$$

This optimization runs over a large number of possible contour configurations - for an image with N pixels, N^K configurations are possible.

4.3.1 Dynamic Programming

Dynamic Programming (DP) [Bellman 1957, Bellman & Dreyfus 1962, Felzenszwalb & Zabih 2011] is as technique commonly used for inference with chain-structured graphical models while Max-Product is its extension that allows handling loopfree graphical models (star- or tree- structured graphs) akin to the models used for human pose estimation [Andriluka *et al.* 2012, Sapp *et al.* 2010, Sapp *et al.* 2011a] and the models used in face and object detection [Felzenszwalb & Huttenlocher 2005, Felzenszwalb *et al.* 2010, Zhu & Ramanan 2012].

Since the merit function in these models is decomposable in pairwise and unary local terms, DP exploits the particular structure of the underlying graph to derive a more efficient algorithm than a brute force search through every possible solution. The computational complexity is then reduced from $\mathcal{O}(N^K)$ to $\mathcal{O}(KN^2)$. To better describe DP, we consider a toy example of a chain graph optimization composed of $K = 3$ nodes as in Figure 4.1. The task is to recover the maximal value of $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3\}$ where each $\mathbf{x}_i \in \mathcal{L}$, $\mathcal{L}$ being the space

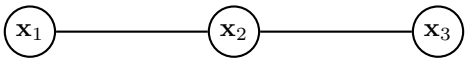


Figure 4.1: A toy example of chain-structured graphical model with 3 nodes.

of possible labels. Applying the commutative property allows us to obtain the following derivations:

$$\begin{aligned}
\omega^* &= \max_{\mathbf{x}_1}\max_{\mathbf{x}_2}\max_{\mathbf{x}_3} S_I(\mathbf{X}) \\
&= \max_{\mathbf{x}_1}\max_{\mathbf{x}_2}\max_{\mathbf{x}_3} \mathcal{U}_{I,1}(\mathbf{x}_1) + \mathcal{U}_{I,2}(\mathbf{x}_2) + \mathcal{U}_{I,3}(\mathbf{x}_3) + \mathcal{P}(\mathbf{x}_1,\mathbf{x}_2) + \mathcal{P}(\mathbf{x}_2,\mathbf{x}_3) \\
&= \max_{\mathbf{x}_1}\max_{\mathbf{x}_2} \mathcal{U}_{I,1}(\mathbf{x}_1) + \mathcal{U}_{I,2}(\mathbf{x}_2) + \mathcal{P}(\mathbf{x}_1,\mathbf{x}_2) + \underbrace{\max_{\mathbf{x}_3} \mathcal{U}_{I,3}(\mathbf{x}_3) + \mathcal{P}(\mathbf{x}_2,\mathbf{x}_3)}_{\mu_3(\mathbf{x}_2)} \\
&= \max_{\mathbf{x}_1} \mathcal{U}_{I,1}(\mathbf{x}_1) + \underbrace{\max_{\mathbf{x}_2} \mathcal{U}_{I,2}(\mathbf{x}_2) + \mathcal{P}(\mathbf{x}_1,\mathbf{x}_2) + \mu_3(\mathbf{x}_2)}_{\mu_2(\mathbf{x}_1)} \qquad (4.6)
\end{aligned}$$

We can see now that the original problem can be solved by combining solutions of nested subproblems expressed by $\mu_3(\mathbf{x}_2)$ and $\mu_2(\mathbf{x}_1)$.

In order to obtain ω^*, we proceed recursively. We solve each subproblem just once and then save its answer in a table. We start by solving the subproblem expressed by $\mu_3(\mathbf{x}_2)$. This allows us to solve the subproblem expressed by $\mu_2(\mathbf{x}_1)$ more easily. What remains then is to solve the problem in Equation 4.6 which is then a simpler problem given the sub-solutions provided by $\mu_2(\mathbf{x}_1)$. Overall, we avoid the work of recomputing the answer every time we solve each subproblem.

4.3.1.1 Generalized Distance Transforms

The bottleneck in inference with dynamic programming is the computation of equations expressed by $\mu_3(\mathbf{x}_2)$ and $\mu_2(\mathbf{x}_1)$, each of which requires N^2 operations to be evaluated. If the label space consists of spacial discretized variables and the pairwise terms $\mathcal{P}(\mathbf{x}_i,\mathbf{x}_j)$ are quadratic, Generalized Distance Transforms (GDTs) [Felzenszwalb & Huttenlocher 2004, Felzenszwalb & Huttenlocher 2005] allow us to compute them in a time linear in the number of pixels. This reduces further the overall complexity of the DP optimization coupled with GDTs from $\mathcal{O}(KN^2)$ to $\mathcal{O}(KN)$.

We recall the expression of $\mu_j(\mathbf{x}_i)$:

$$\mu_j(\mathbf{x}_i) = \max_{\mathbf{x}_j} \mathcal{U}_{I,j}(\mathbf{x}_j) + \mathcal{P}(\mathbf{x}_i,\mathbf{x}_j) \qquad (4.7)$$

[Felzenszwalb & Huttenlocher 2004] show that the problem is equivalent to finding the minimum over a set of offset parabolas arranged in the label space and find this minimum in a linear time complexity.

4.3.2 Shortest Path Algorithms

For convenience of the presentation, we reformulate our maximization problem in Equation 4.8 into an equivalent energy function minimization problem, and write:

$$\mathbf{X}_I^* = \underset{\mathbf{X}}{\operatorname{argmin}} -S_I(\mathbf{X}, \mathbf{w}). \tag{4.8}$$

$$= \underset{\mathbf{X}}{\operatorname{argmin}} \sum_{i=1}^{K} -\mathcal{U}_{I,i}(\mathbf{x}_i) + \sum_{i=1}^{K-1} -\mathcal{P}_i(\mathbf{x}_i, \mathbf{x}_{i+1}). \tag{4.9}$$

This minimization problem can be recast as search for the minimum cost path through a trellis graph. The trellis contains K columns corresponding to the number of landmarks and N rows corresponding to the number of pixels as shown in Figure 4.2 for $K = 3$ and $N = 4$.

We refer to each node with its coordinates $(i, \mathbf{x}_i)$ in the trellis, where $1 \leq i \leq K$ and $\mathbf{x}_i \in \mathcal{L}$. $\mathcal{L}$ is the space of pixel locations (h, v) and we have $|\mathcal{L}| = N$. Each node $(i, \mathbf{x}_i)$ in this trellis corresponds to the assignment of a pixel location $\mathbf{x}_i$ to a landmark i of the chain. We add to this trellis two particular nodes s and t; s is the source node and t is the terminal node.

Edges between the source node s and $(1, \mathbf{x}_1)$ have weight:

$$\mathcal{C}_{s,1}(s, \mathbf{x}_1) = -\mathcal{U}_{I,1}(\mathbf{x}_1). \tag{4.10}$$

Edges between $(i, \mathbf{x}_i)$ and $(i+1, \mathbf{x}_{i+1})$ have weight:

$$\mathcal{C}_{i,i+1}(\mathbf{x}_i, \mathbf{x}_{i+1}) = -\mathcal{U}_{I,i}(\mathbf{x}_{i+1}) - \mathcal{P}_{i,i+1}(\mathbf{x}_i, \mathbf{x}_{i+1}). \tag{4.11}$$

Edges between $(K, \mathbf{x}_K)$ and the terminal node t have weight:

$$\mathcal{C}_{K,t}(\mathbf{x}_k, t) = 0. \tag{4.12}$$

In this trellis a path from s to t traverses exactly one node per landmark in the chain; this corresponds to assigning a pixel to each landmark. The cost/length of a path is the sum of the costs of the edges traversed in the path from s to t. As such, a shortest path from s to t corresponds to a global minimum of the energy function expressed by taking the negative of Equation 4.1. Minimizing this cost function can thus be accomplished by searching for the minimal cost path in a trellis graph. In the following, we review two best-fist search techniques -akin to A^* and Dijkstra's algorithm which are broadly used in the artificial intelligence domain. This will paves the way to describe our accelerating technique described in Section 4.4.

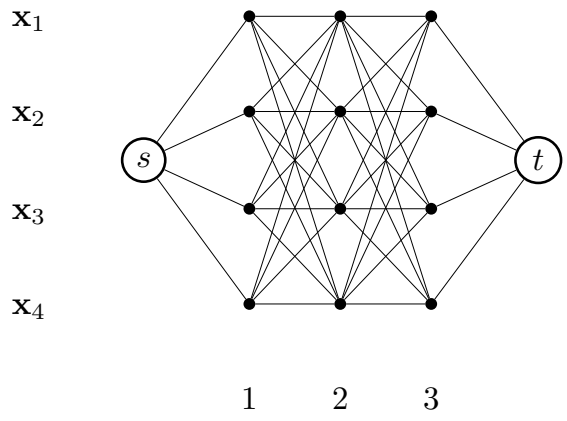


Figure 4.2: A trellis graph. The optimization of Equation 4.1 can be recast as a search for the minimum cost path from s to t in this trellis graph. The trellis shown here contains $K = 3$ columns corresponding to the number of landmarks and $N = 4$ rows corresponding to the number of pixels.

4.3.2.1 Dijkstra's Algorithm

Starting from the source node s, Dijkstra's search algorithm [Dijkstra 1959] builds partial shortest paths $\mathbf{X}_i = \{\mathbf{x}_1, \ldots, \mathbf{x}_i\}, i \leq K$, in order of increasing length. The algorithm explores nodes $(i, \mathbf{x}_i)$ in the trellis according to their shortest path cost $c(i, \mathbf{x}_i)$ from the source node s. We call $c(i, \mathbf{x}_i)$ the *cost-so-far* at node $(i, \mathbf{x}_i)$ defined as:

$$c(i, \mathbf{x}_i) = \min_{\mathbf{x}_1, \ldots, \mathbf{x}_{i-1}} \mathcal{C}_{s,1}(s, \mathbf{x}_1) + \sum_{k=1}^{i-1} \mathcal{C}_{k,k+1}(\mathbf{x}_k, \mathbf{x}_{k+1}) \tag{4.13}$$

$$= \min_{\mathbf{x}_1, \ldots, \mathbf{x}_{i-1}} c(i, \mathbf{x}_{i-1}) + \mathcal{C}_{i-1,i}(\mathbf{x}_{i-1}, \mathbf{x}_i). \tag{4.14}$$

A priority queue $\mathcal{Q}$ guides the exploration of nodes. Each element $\mathcal{E} \in \mathcal{Q}$ is a structure $\mathcal{E} = \{(i, \mathbf{x}_i), P(i, \mathbf{x}_i), (i-1, \mathbf{x}_{i-1})\}$ composed of a trellis node coordinates pair $(i, \mathbf{x}_i)$, an assigned priority $P(i, \mathbf{x}_i)$ and a pointer to a predecessor node $(i-1, \mathbf{x}_{i-1})$.

Initially, $\mathcal{Q}$ is empty. We start by pushing to it the structure composed of the source node s with priority $P(0, s) = 0$ and no predecessor. At each iteration, the element with the lowest priority is popped from $\mathcal{Q}$. If the node corresponding to $(i, \mathbf{x}_i)$ in the trellis was not visited before, we set its path cost to $c(i, \mathbf{x}_i) = P(i, \mathbf{x}_i)$ and we mark it as visited. At the next trellis column $i + 1$ each neighboring node $(i + 1, \mathbf{x}_{i+1})$ is a potential extension of the shortest path arriving to $(i, \mathbf{x}_i)$. The length of this extended path is $c(i, \mathbf{x}_i) + \mathcal{C}_{i,i+1}(\mathbf{x}_i, \mathbf{x}_{i+1})$ according to Equation 4.14. We push to $\mathcal{Q}$ all elements

$\mathcal{E} = \{(i+1, \mathbf{x}_{i+1}), P, (i, \mathbf{x}_i)\}$ where $(i+1, \mathbf{x}_{i+1})$ is a neighbor of $(i, \mathbf{x}_i)$ with a priority set to:

$$P(i+1, \mathbf{x}_{i+1}) = c(i, \mathbf{x}_i) + \mathcal{C}_{i,i+1}(\mathbf{x}_i, \mathbf{x}_{i+1}) \tag{4.15}$$

The priority assigned to each node $(i+1, \mathbf{x}_{i+1})$ is the cost of the shortest path arriving to this node and passing through $(i, \mathbf{x}_i)$. It is hence an estimation of the cost-so-far $c(i+1, \mathbf{x}_{i+1})$.

The algorithm converges when the popped node from $\mathcal{Q}$ is the final node s. We can then recover the shortest path through backtracking; each node in the shortest path indicates its predecessor.

This algorithm finds the global shortest path while if w^* is the cost of the desired shortest path, then Dijkstra ends up visiting nodes such that $c(i, \mathbf{x}_i) \leq w^*$. as proved in [Cormen *et al.* 2001].

4.3.2.2 A^* Search

The A^* search algorithm [Hart *et al.* 1968, Hart *et al.* 1972] can be used to accelerate Dijkstra's algorithm. A^* reduces the number of explored nodes by anticipating the additional least path cost $d(i, \mathbf{x}_i)$ from a node $(i, \mathbf{x}_i)$ to reach the final node t. The 'cost-to-go' $d(i, \mathbf{x}_i)$ at node $(i, \mathbf{x}_i)$ is expressed as:

$$d(i, \mathbf{x}_i) = \min_{\mathbf{x}_{i+1} \dots \mathbf{x}_K} \sum_{k=i}^{K-1} \mathcal{C}_{k,k+1}(\mathbf{x}_k, \mathbf{x}_{k+1}). \tag{4.16}$$

Computing $d(i, \mathbf{x}_i)$ can be demanding, but A^* estimates the cost-to-go through a heuristic function $h(i, \mathbf{x}_i)$. The difference between A^* and Dijkstra's algorithm resides in the priority of the structures. This priority is now set equal to the cost of the path arriving to $\mathbf{x}_{i+1}$ and passing through $\mathbf{x}_i$, plus an estimate of the cost-to-go from $\mathbf{x}_{i+1}$ to terminal node t. This can be written as follows:

$$P(i+1, \mathbf{x}_{i+1}) = c(i, \mathbf{x}_i) + \mathcal{C}_{i,i+1}(\mathbf{x}_i, \mathbf{x}_{i+1}) + h(i+1, \mathbf{x}_{i+1}) \tag{4.17}$$

Due to the introduction of this heuristic, A^* limits the number of explored nodes $(i, \mathbf{x}_i)$ to the set of nodes satisfying:

$$c(i, \mathbf{x}_i) + h(i, \mathbf{x}_i) \leq w^*. \tag{4.18}$$

Given that $h(.) \geq 0$ it becomes clear that this is a smaller set than the set of nodes satisfying Equation 4.3.2.1 visited by Disjkstra's algorithm.

A^* leads to a globally optimal solution if the heuristic is a lower bound on the actual *cost-to-go*:

$$h(i, \mathbf{x}_i) \leq d(i, \mathbf{x}_i), \forall i, \mathbf{x}_i \tag{4.19}$$

In this case, the heuristic is qualified as an *admissible* heuristic. The design of an appropriate heuristic function $h(i, \mathbf{x}_i)$ is the main challenge in devising an efficient A^* algorithm. Ideally the heuristic function should (i) provide a tight lower bound on the *cost-to-go* at a given node $\mathbf{x}_i$, i.e. Equation 4.19 and (ii) be easy to compute. Problem abstraction [Held & Karp 1971, Pearl 1984, Prieditis 1991] is a common way to construct admissible heuristics and consists in relaxing some of the problem constraints.

4.4 Fast Optimization for Chain-structured DCMs

Having covered the basic notions of Dijkstra's algorithm and A^* , we now turn to present algorithms more relevant to our problem. The main challenge here is that we have a large label space that however can be properly manipulated because of its geometric structure since each label corresponds to a position in the image. For this we draw inspiration from recent techniques that were developed in star-shaped DPM detection [Kokkinos 2011a] and adapt them to the case of chain-structured graphs.

4.4.1 Hierarchical A^*

To accelerate inference with chain-structured graphs, we build on the Hierarchical A^* Lightest Derivation (HA^*LD) [Felzenszwalb & Mcallester 2007] algorithm. In the following we briefly review HA^*LD which was originally introduced as a generic method for parsing. Here we proceed to a presentation taylored to our shortest path problem.

HA^*LD is an extension to A^* and uses an hierarchy of problem abstractions to define heuristics. In particular, the algorithm defines problem abstractions by building an hierarchy of coarser versions of the original problem.

Starting from $\mathcal{G}^0$ -the trellis graph defining the original problem- we construct $M = \log_2(N)$ trellis graphs $\mathcal{G}^1 \dots \mathcal{G}^M$, where N is the number of pixels. For convenience, we assume that $N = 2^M$. All the trellis graphs have the same number of columns K but each graph $\mathcal{G}^m$ has fewer nodes at each column than the one below it $\mathcal{G}^{m-1}$. Each trellis graph in this hierarchy represents a problem abstraction to the problem represented by the trellis below it. In particular, each trellis graph $\mathcal{G}^m$ is generated from the trellis below $\mathcal{G}^{m-1}$ in the following manner: at $\mathcal{G}^{m-1}$, each two consecutive nodes $(i, \mathbf{x}_i)$ and $(i, \mathbf{x}'_i)$ at each column i are aggregated to define a *supernode* in the coarser trellis $\mathcal{G}^m$. We refer to this supernode as the *parent* node of the two nodes relying at the finer resolution, and we use pa(.) and ch(.) to denote parent and child

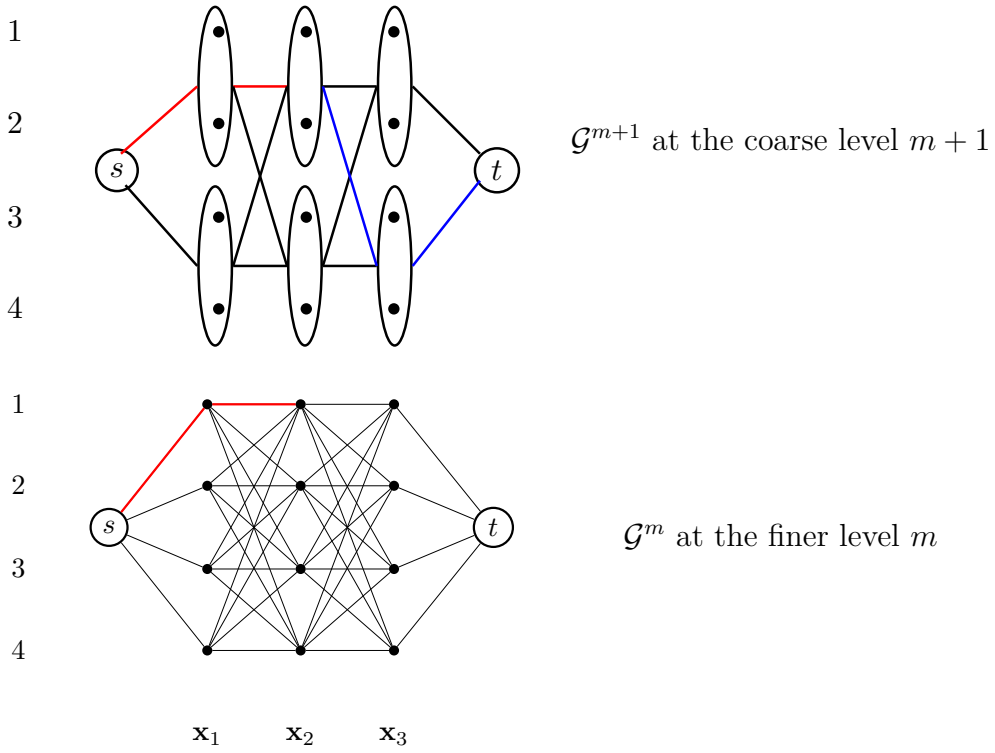


Figure 4.3: Illustration of heuristic computation in the HA^*LD algorithm; the cost-to-go $d^{m+1}(2, \text{pa}(\mathbf{x}_1))$ at the parent node $(m+1, 2, \text{pa}(\mathbf{x}_1))$ in the coarse level $m+1$ serves as a heuristic for node $(m, 2, \mathbf{x}_1)$ at the fine level m of the hierarchy. This cost-to-go is set to lower bound he cost of any path connecting $(m, 2, \mathbf{x}_1)$ to final node t in the finer graph.

operators respectively. In particular, $\text{ch}(\mathbf{x}_i)$ denotes the set of children of a label $\mathbf{x}_i$. Henceforth, we will be denoting each node in this hierarchy with its coordinates $(m, i, \mathbf{x}_i)$, where $0 \leq m \leq M$, $1 \leq i \leq K$ and $0 \leq \mathbf{x}_i \leq \frac{N}{2^m}$. We recall that M is the number of the trellis graphs in the hierarchy, K is the number of landmarks and N is the number of pixels.

At each trellis $\mathcal{G}^m$, each edge cost between nodes $(m, i, \mathbf{x}_i)$ and $(m, i+1, \mathbf{x}_{i+1})$ is set to be a lower bound on the edge costs between all of their possible child nodes $(m-1, i, \text{ch}(\mathbf{x}_i))$ and $(m-1, i+1, \text{ch}(\mathbf{x}_{i+1}))$ at trellis $\mathcal{G}^{m-1}$. This can be written as follows:

$$\mathcal{C}^m_{i,i+1}(\mathbf{x}_i, \mathbf{x}_{i+1}) \leq \min_{\mathbf{x}'_i \in \text{ch}(\mathbf{x}_i), \mathbf{x}'_{i+1} \in \text{ch}(\mathbf{x}_{i+1})} \mathcal{C}^{m-1}_{i,i+1}(\mathbf{x}'_i, \mathbf{x}'_{i+1}); \tag{4.20}$$

Combining Equation 4.20 and Equation 4.16, we can write:

$$d^m(i, \mathbf{x}_i) \leq d^{m-1}(i, \text{ch}(\mathbf{x}_i)), \tag{4.21}$$

where the m superscript indicates the coarsening level for which the cost-to-go is computed. Equation 4.21 allows us to define a heuristic function at any node $(m, i, \mathbf{x}_i)$ in the hierarchy using $d^{m+1}(i, \text{pa}(\mathbf{x}_i))$ -the *cost-to-go* at its parent node lying in the coarser trellis $\mathcal{G}^{m+1}$. Formally, we can now write our heuristic as follows:

$$h^m(i, \mathbf{x}_i) = d^{m+1}(i, \text{pa}(\mathbf{x}_i)). \tag{4.22}$$

The *cost-to-go* $d^{m+1}(i, \text{pa}(\mathbf{x}_i))$ at the parent node $(m+1, i, \text{pa}(\mathbf{x}_i))$ is in itself a shortest path problem from the terminal node t to the current node $(m+1, i, \text{pa}(\mathbf{x}_i))$. Its computation is prioritized. The priority of a *cost-to-go* $d^{m+1}(i, \text{pa}(\mathbf{x}_i))$ is the path cost of the shortest path passing through $(i, \text{pa}(\mathbf{x}_i))$ in the coarse trellis $\mathcal{G}^{m+1}$. Hence, the original problem is solved recursively through the trellis hierarchy.

HA^*LD computes the required heuristics for shortest path search while avoiding exhaustive computation. For this, the algorithm relies on a single global priority queue $\mathcal{Q}$ for all the computations. Starting from the coarsest trellis $\mathcal{G}^M$, HA^*LD progressively refines only the necessary part of the hierarchy and computes heuristics only when needed. This results in an interleaved computation of the cost-to-go and the cost-so-far at nodes in the hierarchy according to the priority of the task.

In order to distinguish between elements in $\mathcal{Q}$ pushed from cost-so-far or cost-to-go computation, we augment each queue element $\mathcal{E} \in \mathcal{Q}$ with a binary variable $\delta \in \{f, b\}$, where f stands for 'forward' and b stands for 'backward'. Each element is then expressed with

$$\mathcal{E} = \{(m, i, \mathbf{x}_i), P_\delta(m, i, \mathbf{x}_i), (m, i-1, \mathbf{x}_{i-1}), \delta\}. \tag{4.23}$$

We note that now the priority depends on δ. If $\delta = f$, we have:

$$P_f^m(i, \mathbf{x}_i) = c^m(i-1, \mathbf{x}_{i-1}) + \mathcal{C}_{i-1,i}^m(\mathbf{x}_{i-1}, \mathbf{x}_i) + d^{m+1}(i, \text{pa}(\mathbf{x}_i)). \tag{4.24}$$

Otherwise, if $\delta = b$, we have:

$$P_b^m(i, \mathbf{x}_i) = d^m(i+1, \mathbf{x}_{i+1}) + \mathcal{C}_{i,i+1}^m(\mathbf{x}_i^m, \mathbf{x}_{i+1}^m) + c^m(i, \mathbf{x}_i). \tag{4.25}$$

The priority of each element in the priority queue explains the use of heuristic functions h and g as depicted by the A^* algorithm; as expressed in Equation 4.24, the priority $P_f(m, i, \mathbf{x}_i)$ of a forward element is the cost-so-far from source node s of the path arriving to $(m, i, \mathbf{x}_i)$ and passing through $(m, i-1, \mathbf{x}_{i-1})$, plus an estimate of the cost-to-go from $(m, i, \mathbf{x}_i)$ to terminal node t. This estimate is computed according to Equation 4.22 and is the cost-to-go from the parent node $(m+1, i, \text{pa}(\mathbf{x}_i))$ to terminal node t.

As expressed in Equation 4.31, the priority $P_b(m, i, \mathbf{x}_i)$ of a backward element is the cost-so-far from terminal node t of the path arriving to $(m, i, \mathbf{x}_i)$ and passing through $(m, i+1, \mathbf{x}_{i+1})$, plus the cost-to-go from $(m, i, \mathbf{x}_i)$ to source node t.

The HA^*LD algorithm is described in more details in Appendix B.

4.4.2 Efficient Computation of Hierarchical Edge Costs

Having described HA^*LD for the shortest path problem, we now turn to describe how we apply this algorithm to solve our specific problem expressed in Equation 4.8. Since our label space consists in pixel locations in the image, we use the KD-tree [Bentley 1975] data structure to partition the search space into a hierarchy of image subspaces. Namely, any node $(m, i, \mathbf{x}_i)$ in our trellis hierarchy -except for nodes lying in the finest trellis- corresponds to a rectangular image subspace populated by nodes $(0, i, \mathbf{x}'_i)$ such that $\mathbf{x}'_i \in \text{ch}^*(\mathbf{x}_i)$. We denote by $\text{ch}^*(.)$ the operator that maps each node to all its descendants lying at the finest trellis. Nodes $(0, i, \mathbf{x}'_i)$ lying at he finest trellis correspond to pixel locations (h, v). In particular, if $\mathbf{x}'_i \in \text{ch}^*(\mathbf{x}_i)$ we have $h \in [l, r]$ and $v \in [b, t]$, where l, r, b, t are the left, right, bottom, top axes defining $\mathbf{x}_i$'s bounding box, $B_{\mathbf{x}_i} = \{l, r, b, t\}$.

The component that is missing from the presentation in Section 4.4.1 is how to compute the edge-costs in the hierarchy of trellis graphs. We recall that we need to compute lower bounds on edge costs between nodes satisfying the constraint expressed by 4.20. In particular, this constraint can be written as follows in our case:

$$\begin{aligned}
\min_{\substack{\mathbf{x}'_i \in \text{ch}^*(\mathbf{x}_i) \\ \mathbf{x}'_{i+1} \in \text{ch}^*(\mathbf{x}_{i+1})}} \mathcal{C}^0_{i,i+1}(\mathbf{x}'_i, \mathbf{x}'_{i+1}) &= \min_{\substack{\mathbf{x}'_i \in \text{ch}^*(\mathbf{x}_i) \\ \mathbf{x}'_{i+1} \in \text{ch}^*(\mathbf{x}_{i+1})}} -\mathcal{U}_{I,i}(\mathbf{x}'_{i+1}) - \mathcal{P}_i(\mathbf{x}'_i, \mathbf{x}'_{i+1}) \\
&\geq \min_{\mathbf{x}'_{i+1} \in \text{ch}^*(\mathbf{x}_{i+1})} -\mathcal{U}_{I,i}(\mathbf{x}'_{i+1}) + \min_{\substack{\mathbf{x}'_i \in \text{ch}^*(\mathbf{x}_i) \\ \mathbf{x}'_{i+1} \in \text{ch}^*(\mathbf{x}_{i+1})}} -\mathcal{P}_i(\mathbf{x}'_i, \mathbf{x}'_{i+1}) \\
&\geq \underline{\mathcal{U}}_{I,i}(\mathbf{x}_{i+1}) + \underline{\mathcal{P}}_i(\mathbf{x}_i, \mathbf{x}_{i+1}) \\
&= \mathcal{C}^m_{i,i+1}(\mathbf{x}_i, \mathbf{x}_{i+1}). \qquad (4.26)
\end{aligned}$$

We compute the first term $\underline{\mathcal{U}}_{I,i}(\mathbf{x}_{i+1})$ through fine-to-coarse minimization when constructing each KD-tree. We compute the second term $\underline{\mathcal{P}}_i(\mathbf{x}_i, \mathbf{x}_{i+1})$ analytically using geometric reasoning; we recall that any node $(m, i, \mathbf{x}_i)$ in the trellis such that $m > 0$ is assigned a rectangular image area delimited by $B_{\mathbf{x}_i} = \{l_{\mathbf{x}_i}, r_{\mathbf{x}_i}, b_{\mathbf{x}_i}, t_{\mathbf{x}_i}\}$. We recall also that our pairwise terms are separable and can be written as:

$$\mathcal{P}_i(\mathbf{x}'_i, \mathbf{x}'_{i+1}) = H(h_i - h_{i+1})^2 + V(v_i - v_{i+1})^2.$$

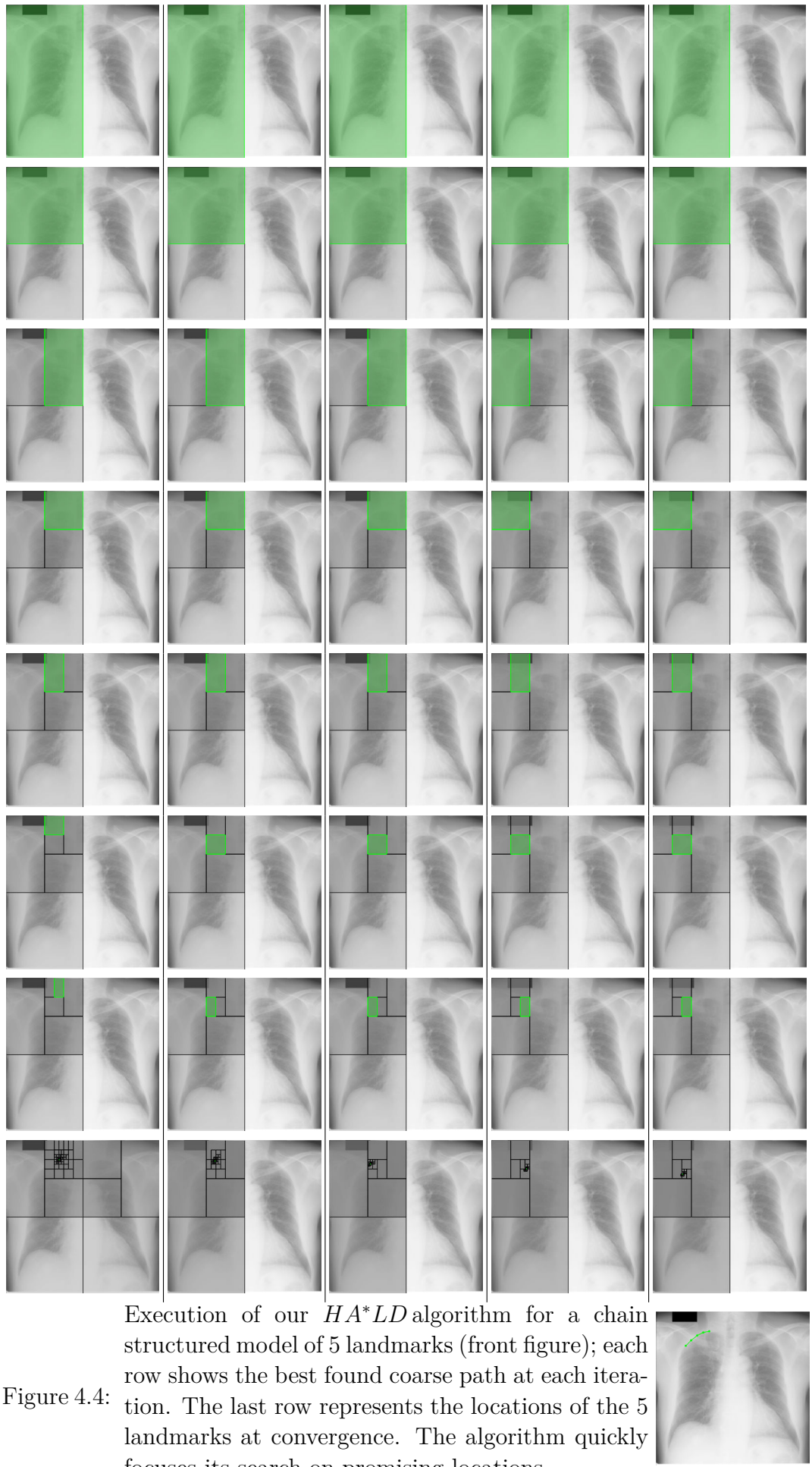


Figure 4.4: Execution of our HA^*LD algorithm for a chain structured model of 5 landmarks (front figure); each row shows the best found coarse path at each iteration. The last row represents the locations of the 5 landmarks at convergence. The algorithm quickly focuses its search on promising locations.

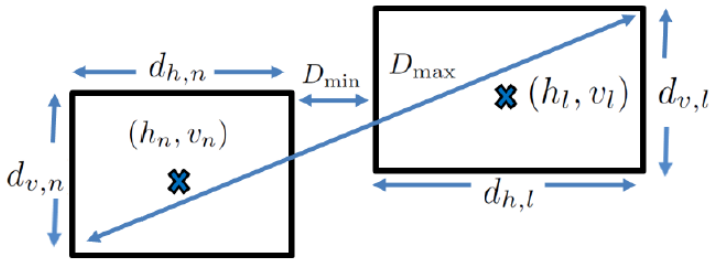


Figure 4.5: Illustration of the terms involved in the geometric bound computations [Kokkinos 2011a].

where $\mathbf{x}'_i = (h_i, v_i), \mathbf{x}'_{i+1} = (h_{i+1}, v_{i+1})$ and we omitted the effect of $\mu_i = (\bar{h}, \bar{v})^T$ in Equation 4.2 for simplicity. This allows us to express $\underline{\mathcal{P}}_i(\mathbf{x}_i, \mathbf{x}_{i+1})$ as:

$$\underline{\mathcal{P}}_i(\mathbf{x}_i, \mathbf{x}_{i+1}) \leq \min_{h_i, v_i} \min_{h_{i+1}, v_{i+1}} H(h_i - h_{i+1})^2 + V(v_i - v_{i+1})^2 \tag{4.27}$$

where $h_i \in [l_{\mathbf{x}_i}, r_{\mathbf{x}_i}]$, $v_i \in [b_{\mathbf{x}_i}, t_{\mathbf{x}_i}]$, $h_{i+1} \in [l_{\mathbf{x}_{i+1}}, r_{\mathbf{x}_{i+1}}]$ and $v_{i+1} \in [b_{\mathbf{x}_{i+1}}, t_{\mathbf{x}_{i+1}}]$. To avoid confusion in the notation, let's denote by B_l the bounding box corresponding to node $(m, i, \mathbf{x}_i)$ and by B_n the bounding box corresponding to node $(m, i+1, \mathbf{x}_{i+1})$ and describe a bounding box B_n using its center $c = (h_n, v_n)$, and dimension $d_{h,n}, d_{v,n}$ attributes as in [Kokkinos 2011a]. We use $\overline{d_h} = \frac{1}{2}(d_{h,n} + d_{h,l})$, $\overline{d_v} = \frac{1}{2}(d_{v,n} + d_{v,l})$ and write:

$$\begin{aligned} \underline{\mathcal{P}}_i(\mathbf{x}_i, \mathbf{x}_{i+1}) &= H \max(\lceil h_n - h_l - \overline{d}_h \rceil, \lceil h_l - h_n - \overline{d}_h \rceil)^2 \\ &+ V \max(\lceil v_n - v_l - \overline{d}_v \rceil, \lceil v_l - v_n - \overline{d}_v \rceil)^2 \end{aligned} \tag{4.28}$$

where $\lceil x \rceil = \max(x, 0)$. We visually depict this bound in Figure 4.5. The lower bound is zero if the boxes overlap, or else equals the distance of their closest points scaled with H and V. The main gain is that instead of evaluating a quantity on the cross product of pixels $\mathbf{x}'_i \in \text{ch}^*(\mathbf{x}_i)$ and $\mathbf{x}'_{i+1} \in \text{ch}^*(\mathbf{x}_{i+1})$, we use two terms that are computable in a number of operations independent of the cardinalities of the respective bounding boxes.

4.4.3 Path Pruning

We argue in this section how we further accelerate our HA^*LD based shortest path algorithm by drawing inspiration from the pruning techniques employed in the branch-and-bound based algorithm of [Kokkinos 2011b]. This requires the combination of upper bounds and lower bounds on the energy ω^* of desired solution so as to eliminate solutions that are guaranteed to be suboptimal.

The goal of pruning is to permanently discard unpromising paths from an early stage of the search. We denote by $\overline{S}(m,i,\mathbf{x}_i)$ an upper bound on the path cost of the shortest path passing by node $(m,i,\mathbf{x}_i)$ and by $\underline{S}(m,i,\mathbf{x}_i)$ a lower bound on the path cost of the shortest path passing by node $(m,i,\mathbf{x}_i)$. The main idea of pruning is that if $\underline{S}(m,i,\mathbf{x}_i) \geq \overline{S}(m,i,\mathbf{x}'_i)$, then the best path passing through node $(m,i,\mathbf{x}_i)$ cannot be better than the worst path passing through $(m,i,\mathbf{x}'_i)$; we do not need then to entertain node $(m,i,\mathbf{x}_i)$ as an hypothesis.

Specifically, we recall that each time we compute a forward priority of a priority queue element $\mathcal{E} = \{(m,i+1,\mathbf{x}_{i+1}), P, (m,i,\mathbf{x}_i), forward\}$, we are actually estimating a lower bound $\lambda^{(m,i,\mathbf{x}_i)}_{(m,i+1,\mathbf{x}_{i+1})}$ on any path passing through nodes $(m,i+1,\mathbf{x}_{i+1})$ and $(m,i,\mathbf{x}_i)$. We consider now that at the time of computing this quantity, we have at our disposal an upper bound μ on the shortest path ω^*. This $\mu \geq \omega^*$ is computed from previous iterations. This allows us to use the following rationale: if the lower bound $\lambda^{(m,i,\mathbf{x}_i)}_{(m,i+1,\mathbf{x}_{i+1})}$ is greater than the available upper bound μ, we deduce that any path passing through $\text{ch}(m,i,\mathbf{x}_i)$ and $\text{ch}(m,i+1,\mathbf{x}_{i+1})$ is a suboptimal candidate solution. Recursively, this is valid for any descendant of $(m,i,\mathbf{x}_i)$ and $(m,i+1,\mathbf{x}_{i+1})$ at any level of the hierarchy. This is because as we go from a coarse trellis to a finer one, lower bounds get higher and upper bounds get lower. We thus prune all these candidates solution from search. This allows us to keep limited the number of terms involved in the computation in finer trellis graphs. Therefore, we dynamically define the structure of the abstraction hierarchy. We find the neighbors $(m,i+1,\mathbf{x}_{i+1})$ of a node $(m,i,\mathbf{x}_i)$ by looking to $(m+1,i+1,\mathbf{x}_{i+1})$ that survived the pruning and considering their children. The pruning allows us as well to reduce the number of elements to be pushed to the priority queue $\mathcal{Q}$. This is actually a costly operation since its computational complexity is $\mathcal{O}(n)$, where n is the number of nodes already in $\mathcal{Q}$.

Now we show how to compute upper bounds on ω^*. To ensure that we prune as many elements as possible, we need to compute tight upper bounds. To this end we start by defining the quantity $\overline{\mathcal{C}}^m_{i,i+1}(\mathbf{x}_i,\mathbf{x}_{i+1})$ that upper bounds all edge costs $\mathcal{C}^0_{i-1,i}(\mathbf{x}'_{i-1},\mathbf{x}'_i)$ for all $\mathbf{x}'_i \in \text{ch}^*(\mathbf{x}_i), \mathbf{x}'_{i+1} \in \text{ch}^*(\mathbf{x}_{i+1})$ For this, we use the same technique used to find compute $\mathcal{C}^m_{i,i+1}(\mathbf{x}_i,\mathbf{x}_{i+1})$ in Equation 4.26.

Namely we write:

$$\begin{aligned}\max_{\substack{\mathbf{x}'_i\in \mathrm{ch}^*(\mathbf{x}_i)\\ \mathbf{x}'_{i+1}\in \mathrm{ch}^*(\mathbf{x}_{i+1})}} \mathcal{C}^0_{i-1,i}(\mathbf{x}'_{i-1},\mathbf{x}'_i) &= \max_{\substack{\mathbf{x}'_i\in \mathrm{ch}^*(\mathbf{x}_i)\\ \mathbf{x}'_{i+1}\in \mathrm{ch}^*(\mathbf{x}_{i+1})}} -\mathcal{U}_{I,i}(\mathbf{x}'_{i+1}) - \mathcal{P}_i(\mathbf{x}'_i,\mathbf{x}'_{i+1})\\ &\leq \max_{\mathbf{x}'_{i+1}\in \mathrm{ch}^*(\mathbf{x}_{i+1})} -\mathcal{U}_{I,i}(\mathbf{x}'_{i+1}) + \max_{\substack{\mathbf{x}'_i\in \mathrm{ch}^*(\mathbf{x}_i)\\ \mathbf{x}'_{i+1}\in \mathrm{ch}^*(\mathbf{x}_{i+1})}} -\mathcal{P}_i(\mathbf{x}'_i,\mathbf{x}'_{i+1})\\ &\leq \overline{\mathcal{U}}_{I,i}(\mathbf{x}_{i+1}) + \overline{\mathcal{P}}_i(\mathbf{x}_i,\mathbf{x}_{i+1})\\ &= \overline{\mathcal{C}}^m_{i,i+1}(\mathbf{x}_i,\mathbf{x}_{i+1}). \end{aligned} \tag{4.29}$$

We compute the first term $\overline{\mathcal{U}}_{I,i}(\mathbf{x}_{i+1})$ through fine-to-coarse maximization when constructing each KD-tree. We compute the second term $\overline{\mathcal{P}}_i(\mathbf{x}_i,\mathbf{x}_{i+1})$ analytically using geometric reasoning. This allows us to write:

$$\overline{\mathcal{P}}_i(\mathbf{x}_i,\mathbf{x}_{i+1})) = H\max(h_n-h_l+\overline{d_h}, h_l-h_n+\overline{d_h})^2+V\max(v_n-v_l+\overline{d_v}, v_l-v_n+\overline{d_v})^2$$

This upper bound is visually depicted in Figure 4.5; the upper bound equals the distance of the further points of the two boxes scaled with H and V. By replacing $\mathcal{C}^m_{i,i+1}(\mathbf{x}_i,\mathbf{x}_{i+1})$ by $\overline{\mathcal{C}}^m_{i,i+1}(\mathbf{x}_i,\mathbf{x}_{i+1})$, we derive an upper bound $\overline{c}^m(i,\mathbf{x}_i)$ on the cost-so-far and an upper bound $\overline{d}^m(i,\mathbf{x}_i)$ on the cost-to-go at node $(m,i,\mathbf{x}_i)$. Hence, at the time of computing priorities for forward elements to be pushed to $\mathcal{Q}$, we compute upper bounds $\overline{P}^m_f(i,\mathbf{x}_i)$ on the best path cost in the following manner:

$$\overline{P}^m_f(i,\mathbf{x}_i) = \overline{c}^m(i-1,\mathbf{x}_{i-1}) + \overline{\mathcal{C}}^m_{i-1,i}(\mathbf{x}_{i-1},\mathbf{x}_i) + \overline{d}^{m+1}(i,\mathrm{pa}(\mathbf{x}_i)) \tag{4.30}$$

Similarly, when computing priorities for backward elements to be pushed to $\mathcal{Q}$, we compute upper bounds $\overline{P}^m_b(i,\mathbf{x}_i)$ on the best path cost as follows:

$$P^m_b(i,\mathbf{x}_i) = \overline{d}^m(i+1,\mathbf{x}_{i+1}) + \overline{\mathcal{C}}^m_{i,i+1}(\mathbf{x}^m_i,\mathbf{x}^m_{i+1}) + \overline{c}^m(i,\mathbf{x}_i). \tag{4.31}$$

As the algorithm advances we keep track of the smallest upper bound encountered so far.

4.4.4 Multi-scale Optimization

To accommodate global scale changes, we run the inference over multiple scales. Namely, an image pyramid is built from the original image I by re-sampling at a set of scales $\mathcal{S} = (1, r, r^2, \ldots, r^S)$ as shown in Figure 4.4.4. The minimum energy contour over these images is recovered:

$$\mathbf{X}^*_{I,\mathcal{S}} = \operatorname*{argmin}_{\mathbf{X},i} s_{I(r^i)}(\mathbf{X}), \tag{4.32}$$

where $I(r^i)$ denotes the image re sampled with a ratio r^i.

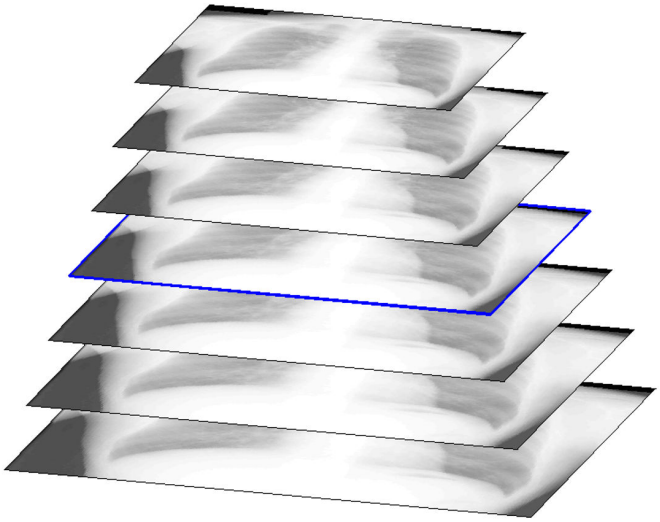


Figure 4.6: In order to accommodate global scale changes, we build an image pyramid and recover the best scoring contour over these images

Similarly to [Kokkinos 2011b], we extend our algorithm to deal with multiple scales. We could run our optimization algorithm separately over multiple scales, but additional time can be saved by prioritizing the search over scales by inserting in the same priority queue partial solutions coming from multiple scales. This allows us to avoid computation at irrelevant scales from an early stage.

4.5 Fast Optimization of the Slave Problems

In this section, we focus on accelerating the optimization of the slave problems in the ADMM based optimization of loopy graphs. Although the slaves are still chain structured, their energy functions differ from the energy function described in Chapter 4. Namely, we now have extra terms that appear in shared nodes between slaves $\mathbf{X}(r)$:

$$\mathbf{X}_I^* = \underset{\mathbf{X}}{\operatorname{argmin}}\, S(\mathbf{X}) + \left(\sum_{r \in R} \lambda(r)\mathbf{X}(r) - \rho \sum_{r} (\mathbf{X}(r) - u(r))^2 \right) \tag{4.33}$$

where r is the point index belonging to more than a slave and $\lambda(r)$ is a Lagrange multiplier and $u(r)$ ensures consistency at overlapping points. The extra term is expressed in the form of a parametric quadratic function $(\mathbf{X}(r) - u(r))^2$ of position.

One straightforward approach is to evaluate this parametric function over all pixel locations and add its values to each candidate solution in the opti-

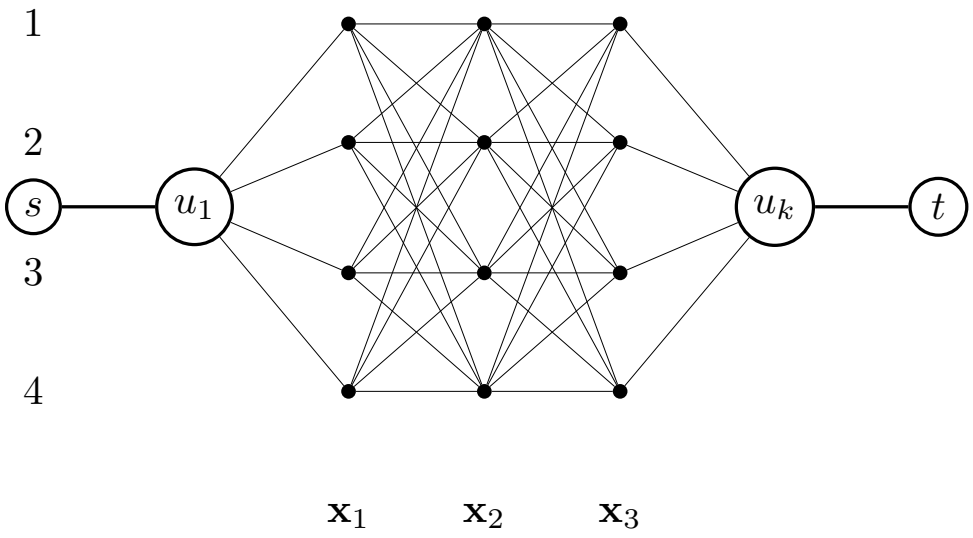


Figure 4.7: Trellis graph to represent the optimization of a slave problem with first and last nodes are shared with other graphs. Two nodes μ_1 and μ_k are added to the trellis to account for the additional term $(\mathbf{X}(r) - u(r))^2$ in Equation 4.33.

mization. This computation can be accelerated by integrating the evaluation of this term in the hierarchical HA^*LD shortest path computation. This allows us to minimize $(\mathbf{X}(r) - u(r))^2$ in a coarse-to-manner without having to evaluate its value over all possible pixel locations. To this end, we modify the trellis hierarchy to account for new virtual nodes that represent these extra terms. An example of a trellis at the finest level of the hierarchy is shown in Figure 4.7, where in that case the first and last nodes of the corresponding chain graphical model are shared with other chains.

In this trellis, edges between nodes $\mathbf{x}_r$ and the node u_r have weight $\mathcal{C}(\mathbf{x}_r, u_r) = (\mathbf{x}_r - u_r)^2$. Starting from this finest resolution, we build the hierarchy of increasingly coarser trellis graphs. Edges in the coarse graphs are computed as follows:

$$\mathcal{C}^m(\mathbf{x}_r, u_r) \leq \min_{\mathbf{x}'_r \in \mathrm{ch}^*(\mathbf{x}_r)} \mathcal{C}(\mathbf{x}'_r, u_r) = \min_{\mathbf{x}'_r \in \mathrm{ch}^*(\mathbf{x}_r)} (\mathbf{x}_r - u_r)^2 \tag{4.34}$$

4.6 Results

We consider the GDTs based optimization as our baseline in order to assess the performance of our algorithm. We ran experiments on a standard PC machine equipped with an $Intel^{®}$ Xeon processor running at 2.66GHz and

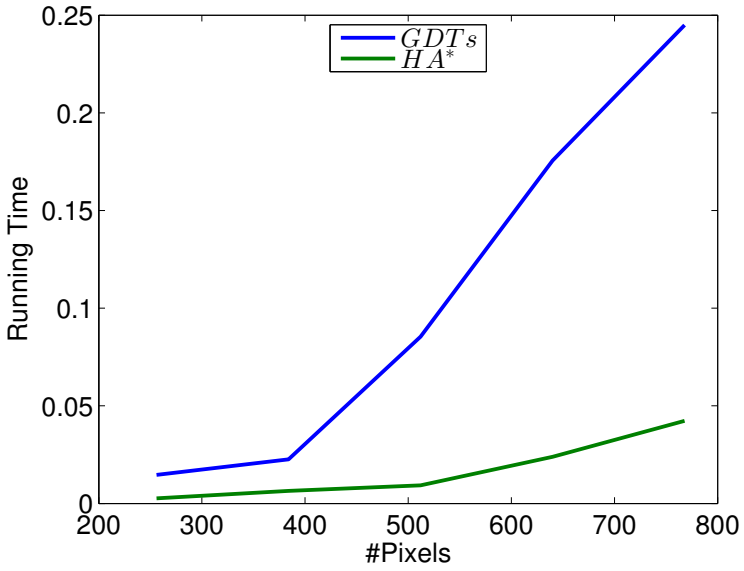


Figure 4.8: GDTs optimization vs A^* running time comparison while varying the image size

12Gb of RAM on $Windows^{®}$ 7. We use a single core in all computations.

We use the publicly available dataset of [Shiraishi *et al.* 2000, van Ginneken *et al.* 2006] which contains 247 standard posterior anterior chest radiographs of healthy and non-healthy subjects (presenting nodules). The database contains segmentations from radiologists, which provide a delineation of the lung fields, the heart and the clavicles as shown in Figure 2.4.1. Gold standard segmentation masks are hence available as well as corresponding landmark positions lying on the contour. All the reported running times are in seconds and averaged over all images belonging to the considered database.

Our experiments show that our HA^*LD-based shortest path algorithm outperforms the optimization based on GDTs in terms of efficiency. We start our side by side comparisons by comparing the single scale optimization of two algorithms.

In the first experiment, we run the algorithms on images with different sizes as shown in Figure 4.6. We verify that our algorithm has increasingly better performance that the GDTs optimization as the number of pixels increases.

In the second experiment, we vary the number of landmarks by subsampling over the landmarks forming the contour as shown in Figure 4.6. We observe that the performance decreases when the number of landmarks increases. This is anticipated since the accumulation of many loose bounds can result in very loose overall bounds.

number of image scales	1	3	5	7	9	11
GDTs	0.027	0.082	0.128	0.192	0.245	2.880
HA^*LD with pruning	0.016	0.025	0.050	0.086	0.113	0.129
HA^*LD without pruning	0.021	0.034	0.060	0.099	0.130	0.139

Table 4.1: Average computational time comparison in seconds.

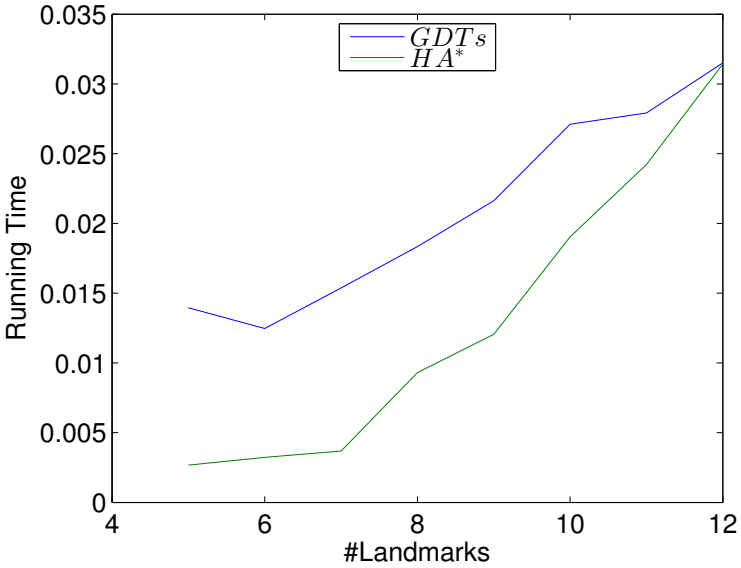


Figure 4.9: GDTs optimization vs A^* running time comparison varying the number of landmarks

In table 4.1, we compare our multi-scale implementation against the multi-scale implementation of GDTs according to the number of scales considered. In these experiments, we consider $K = 10$ as the number of landmarks. We observe that our gain margin increases as the number of scales increases. This is due to the use of a single global priority queue. Moreover, we show in this table the benefits of the pruning technique by comparing running times with a version of HA^*LD without pruning.

A number of qualitative results on segmentation of the right lung and left lung are shown in Figure 4.10. The learning part for this open chain structured model will be described in Chapter 2.

4.7 Conclusion

In this chapter, we proposed an efficient algorithm for chain-structured graphs with a large label space. Our approach is based on the HA^*LD algorithm aug-

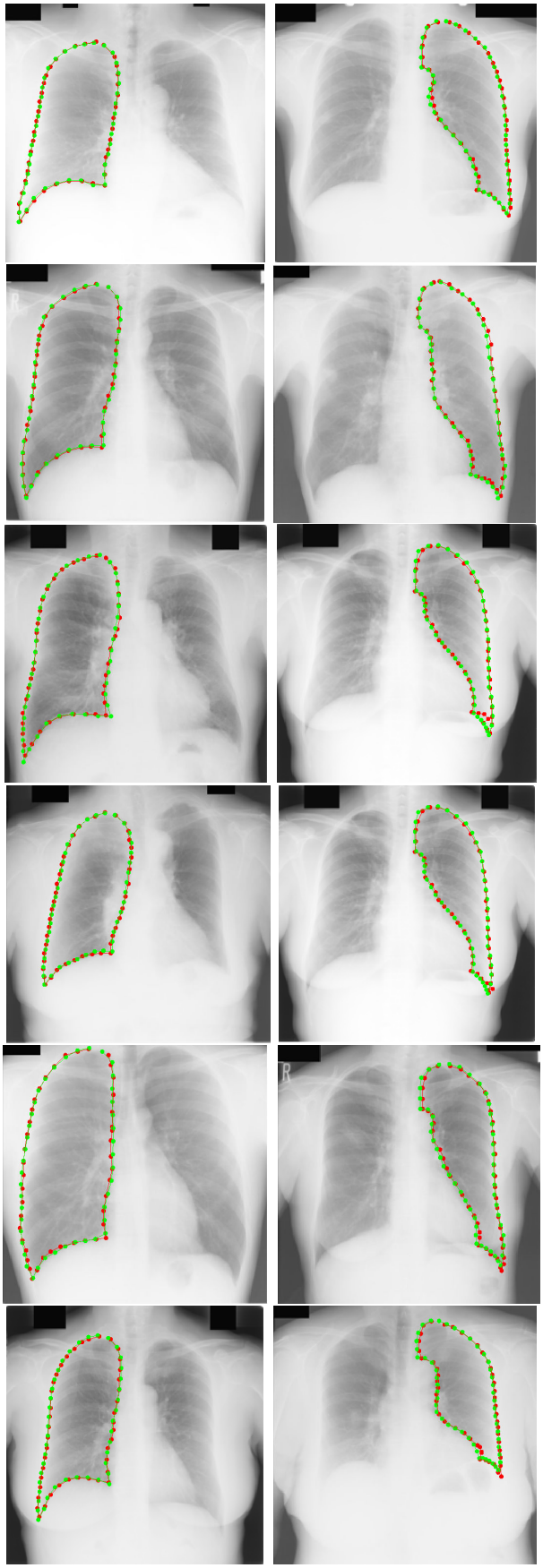


Figure 4.10: Segmentation results on the left lung and on the right lung using our open chain-structured deformable contour model. Ground truth contours are shown in green, our results are shown in red.

mented with a pruning technique -used in branch and bound algorithms- which further accelerates the optimization. We applied our algorithm on the task of shape segmentation of medical images where we showed up to 23-fold speedup in multi-scale optimization comparing to the state of the art technique based on dynamic programming accelerated with generalized distance transforms.

Although we have focused on the shape segmentation application, we view the main contribution of this paper as providing a fairly generic method for efficient inference on chain structured graphical models with spatial variables. Our method can be easily applied to face recognition and body pose estimation tasks to accelerate the optimization. Unlike the GDTs based optimization which requires separable kernels, our method is more flexible regarding the form of the pairwise potentials. This can be done by properly modifying the lower and upper bounds computation on the edge costs.

Chapter 5

Conclusions and Future Work

Contents

5.1 Introduction

The purpose of this work has been to improve deformable part models for shape-based segmentation in medical images. In this thesis, we derived efficient algorithms to solve optimization problems that arise in deformable part models. This allowed us to use richer graphs with arbitrary topologies, optimize loss functions specific to contours, and segment objects in medical images faster that the state-of-art techniques.

5.2 Our Contributions

Below we summarize our contributions on three aspects of the shape segmentation problem: modeling, inference and learning.

Modeling

We cast multi-organ shape segmentation and landmark localization in a graphical model framework. In particular, we represented every organ as a cyclic graph, whose nodes indicate landmarks positions and we used loopy graphs to incorporate problem constraints that cannot be encoded through chain- or tree- structured graphs. We integrate dense local descriptors in the appearance terms to achieve better performance than convolution-based features currently used in medical image analysis.

Inference

We derived an efficient algorithm to tackle the inference problem underlying DPM-based shape segmentation. We employed ADMM to decompose the associated loopy graph into open chain-structured subgraphs. ADMM

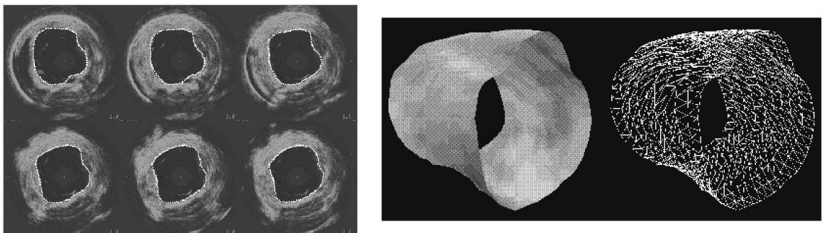


Figure 5.1: The surface of a 3D object can be decomposed into chains belonging to horizontal, sagittal and coronal slices sharing 3D landmarks.

allowed us to fix the potential inconsistencies of the individual solutions in a small number of iterations (typically ≤ 10). We optimized efficiently each of the subgraph problems -that arose from the ADMM decomposition- in a coarse-to-fine approach to solve shortest path problems. Specifically, we adapted HA^*LD and accelerated it through pruning techniques and a multi-scale efficient implementation.

Learning

We used structural SVMs to learn jointly all the model parameters and designed the structured MCD loss function which is suited to the task of shape segmentation. The resulting training algorithm inherits efficiency from our inference algorithm since an adapted version of the latter is used as a subroutine in learning with the Cutting Plane algorithm. Training with Structured Prediction and the MCD loss delivered a superior performance over structured prediction with the generic 0-1 loss, which already outperforms the state-of-the-art method, according to our experiments on the considered medical benchmark.

Overall, the use of our fast and exact algorithms almost saturated the considered benchmark while being 10 times faster in average over the established state-of-the-art.

5.3 Future Work

We now discuss several directions of research that can be investigated to improve and extend the presented work.

Application

On the application level, since our method is fairly generic, the most straightforward extension is to transfer our method to the segmentation of anatomical structures in 3D volumes. There, the object's surface can be decomposed into 2D chains -as shown in Figure 5.1- on sagittal, horizontal and

coronal slices that share nodes. Since a large number of landmarks is required to represent the 3D surface of the object, the manual labeling of the training shapes to establish correspondences between landmarks, becomes rapidly a tedious task. Such datasets are currently rare for 3D medical applications.

The application of our method to other challenging vision tasks including face recognition and pose estimation is also conceivable. A main challenge in face detection using ADMM inference is that many object instances may exist in the same image and the slaves might then be attracted by object parts that belong to different objects.

Modeling

Regarding the modeling, we note that in the considered dataset, all the training shapes have similar orientations due to acquisition. This might not be the case for other applications. Hence a rotation invariant model is required for an accurate detection of revolving objects in images for instance. To this end, we can think of combining the rotation invariant appearance features, SID of [Kokkinos & Yuille 2008] with the weak pictorial structure model of [Gu 2012], where rotation invariant pairwise potentials are designed.

Another possible extension to our model is to consider image region support in order to improve the quality of the segmentation. The region inside the anatomical object can be triangulated as shown in Figure 5.2. Then Green's theorem allows to express regional integrals as inner products on the local 2D line segments represented by pairwise terms in the energy function.

Inference

On the inference side, the adaptation of our inference algorithm to pairwise terms encoding region statistics and/or rotation invariance is challenging. To this end, one direction is to precompute pairwise terms for all possible values and cache them in a database. We also wish as to explore ADMM-based optimization for higher-order MRF optimization, used e.g. to encode symmetry or hypergraph optimization.

Learning

On the learning side, we considered in this work the optimization of the MCD loss. A different performance measure can be considered to improve performance. In particular, the optimization of the widely used DICE coefficient is appealing. This requires solving efficiently a new loss-augmented inference problem.

In our structured prediction learning algorithm, we used ADMM as a black box optimization method to solve the augmented inference subroutine need by the cutting plane algorithm. Therefore, ADMM was called many times during learning. [Komodakis 2011, Hazan & Urtasun 2010] propose to reduce training of complex graphs to parallel training of a series of easy-to-handle slave graphs by using a dual decomposition framework. Based on this idea,

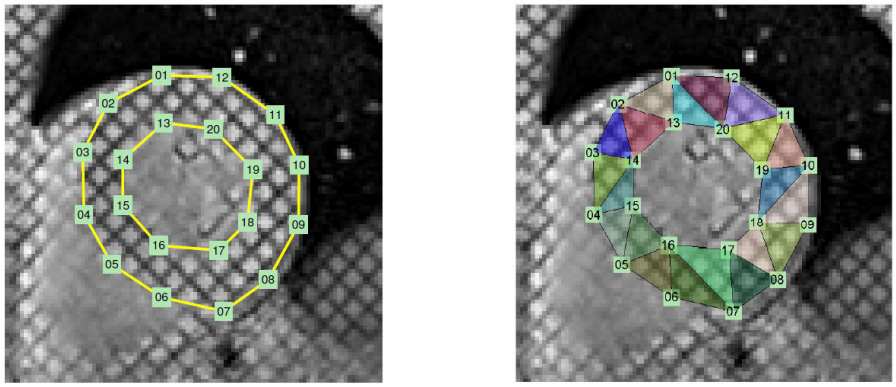


Figure 5.2: Model triangulation from [Xiang *et al.* 2013].

we can think of using ADMM instead of dual decomposition to optimize the training objective and aim for faster convergence of the learning algorithm.

Furthermore, since we now have at our disposal an efficient inference algorithm that makes no assumption about the graph topology, we can think about the optimal structure of the graph. For instance, we can investigate if longer connections between landmarks would improve the model performance -without worrying about the resulting model topology. To this end, the use of supervised machine learning techniques to predict the optimal topology of the graph is worth considering.

Appendix A

Rapid Mode Estimation for 3D Brain MRI Tumor Segmentation

Contents

In this Appendix we develop a method to rapidly initialize 3D brain tumor segmentation; tumor segmentation is crucial for brain tumor resection planning, and a high-quality initialization can significantly impact segmentation quality. Our main contribution lies in developing an efficient method to initialize the segmentation by phrasing it as nonparametric density mode estimation, and developing a Branch and Bound method to efficiently find the mode (maximum) of the density function.

Our technique is exact, has guaranteed convergence to the global optimum, and scales logarithmically in the volume size by virtue of recursively subdividing the search space. Our method employs the Dual Tree data structure originally developed for kernel density estimation [Gray 2003], and recently used for object detection with branch-and-bound in [Kokkinos 2011a]. In this

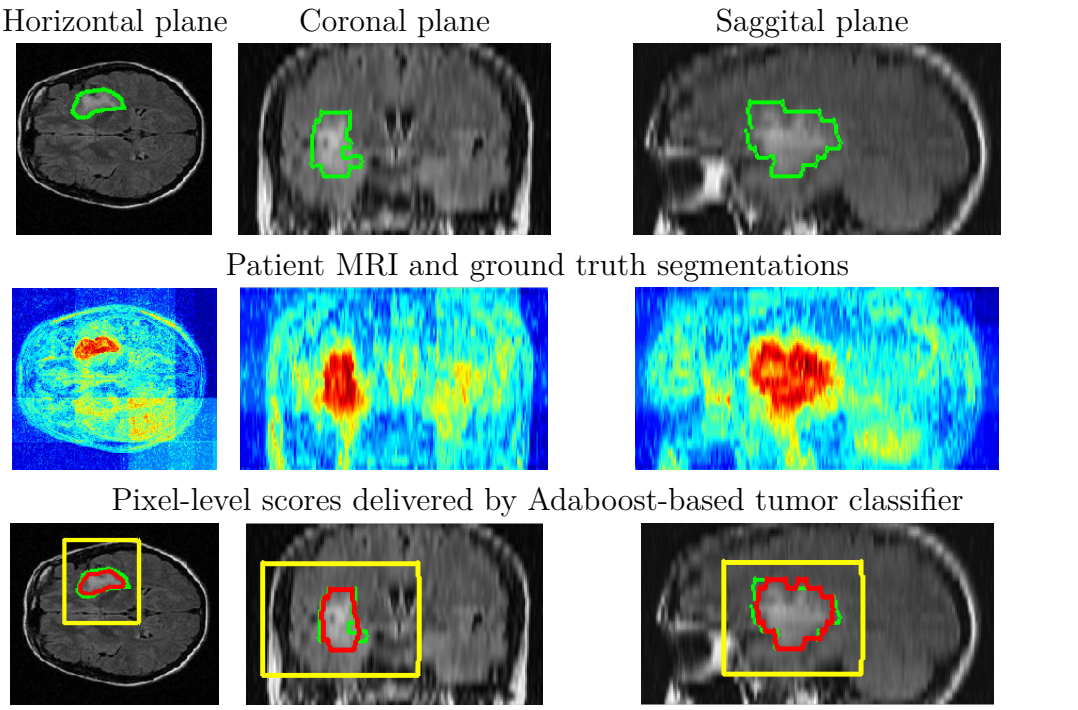


Figure A.1: Illustration of our method: we use ground truth annotations to train pixel-level tumor classifiers using boosting. The pixel-level classifier scores are treated as weights in kernel density estimation KDE; the mode of the resulting KDE can be interpreted as the center of the tumor. We use branch-and-bound to rapidly find the mode of the KDE (yellow box); this is used to initialize the graph-cut segmentation that delivers the contours shown in red.

work we 'close the loop', and use the Dual Tree data structure to find the mode of a nonparametric distribution.

This estimated mode provides our system with an initial tumor hypothesis; this is then refined by graph-cuts to provide a sharper outline of the tumor area. We demonstrate a 12-fold acceleration with respect to a standard mean-shift implementation, allowing us to accelerate tumor detection to a level that would facilitate high-quality brain tumor resection planning.

A.1 Introduction

The most common treatment of brain tumors is surgical resection, where the quality of the intervention can be largely affected by the efficient iden-

tification of the surgical margins during planning. Conventional segmentation techniques rely on prior knowledge and smoothness constraint to overcome the enormous variability of tumors both in terms of location as well as in terms of geometric characteristics. Even though recent studies [Duffau 2004] indicate statistically preferable locations for tumors in the brain and [Parisot *et al.* 2012] proved that using this information improves substantially the results, in our work we take a more agnostic approach, using a clustering-based method for tumor detection.

Clustering, segmentation and nonparametric density mode estimation are related problems whose combination has been particularly studied in 2D images in the thread of works developed around the Mean-Shift method [Comaniciu & Meer 2002]. This method is also used as a component in a number of diagnosis tools such as vessel tracking [Walsum *et al.* 2008], Multiple Sclerosis brain segmentation [Prima *et al.* 2008] and MRI brain clustering [Mayer & Greenspan 2009], and is a general tool that applies transversally to a host of problems in medical imaging.

In our work we develop a method to rapidly initialize a segmentation by relying on nonparametric density mode estimation. The mode of a nonparametric density estimate is used to pinpoint the center of the tumor, and thereby initialize a 3D segmentation implemented using graph-cuts. Our main contribution lies in addressing the computational complexity of the mode estimation problem.

The original Mean Shift method [Comaniciu & Meer 2002] is iterative, scales linearly in the number of points used in the Kernel Density Estimation (KDE) (as it follows the trajectory of every one of them) and can require careful checking of convergence. Faster variants of Mean Shift exist including Medoid Shift [Sheikh *et al.* 2007], Quick Shift [Vedaldi & Soatto 2008], Fast Gauss transforms [Yang *et al.* 2003] as well as the Dual Tree variant of Mean Shift [Wang *et al.* 2007]. However, those of them that are exact [Yang *et al.* 2003, Wang *et al.* 2007] are 'dense' i.e. evaluate the KDE over a dense set of locations; as such they may be inappropriate for application to 3D medical image volumes. Alternatively, those that focus on the modes [Sheikh *et al.* 2007, Vedaldi & Soatto 2008] are only approximate and have complexity proportional to $O(KN)$ where N is the number of pixels and K is the typical neighborhood size.

The main contribution of our work is a rapid mode estimation technique for 3D MRI image segmentation. Dealing with three dimensional data challenges several algorithms which are reasonably efficient for 2D medical image analysis. In this paper, we leverage upon recent developments using Branch-and-Bound (BB) in object detection [Kokkinos 2011a, Kokkinos 2012b, Kokkinos 2012a], which demonstrated that detection is pos-

sible in time sub-linear in the image size.

The main thrust of our work is the adaptation of this idea to the mode finding problem in KDE, typically addressed through Mean Shift. We propose an algorithm that can find the mode of the density with best-case complexity being logarithmic in the size of the search space.

We apply our algorithm to the setting of 3D brain tumor segmentation. Our algorithm takes the scores of a discriminatively trained classifier for tumor voxels and uses them to construct weights for a KDE-based estimate of the tumor location. Using standard mean shift would require tracking the trajectory of each voxel, and identifying the largest basin of attraction. Instead our algorithm narrows down the location of the maximum through an iterative branch-and bound algorithm. In specific, we construct bounds on the value of the KDE over intervals of the search space, and use these bounds to devise a prioritized search strategy for the density's mode. We demonstrate substantial speedups when comparing to the standard mean-shift algorithm.

Furthermore, we couple the mode estimation results with a post-processing step using graph-cuts, which allows us to boost the segmentation performance of our algorithm. Systematic evaluations are performed on clinical datasets demonstrating a 12-fold acceleration in speed over classical Mean-Shift while at the same time achieving robust tumor detection and segmentation.

A.2 Rapid 3D Structure Detection

Our goal is to devise an algorithm that can quickly detect the largest region corresponding to a class (tumor in our case) within a 3D volume. This problem is particularly challenging for standard segmentation algorithms as it is hard to define exact boundaries between tumor and normal tissue [Birkbeck *et al.* 2009]. Moreover, relying only on a classifier to separate the tumor class from the remaining structures in the MRI volume is challenging, due to the similarity between tumor and normal tissue and the high diversity in appearance of tumor tissue among different patients [Birkbeck *et al.* 2009].

We start by phrasing our problem as mode seeking for a Kernel Density Estimate and then proceed to describing our Branch-and-Bound based optimization algorithm. We note that even though we focus on tumor segmentation, the same approach could potentially be useful to other maximization problems in 3D space.

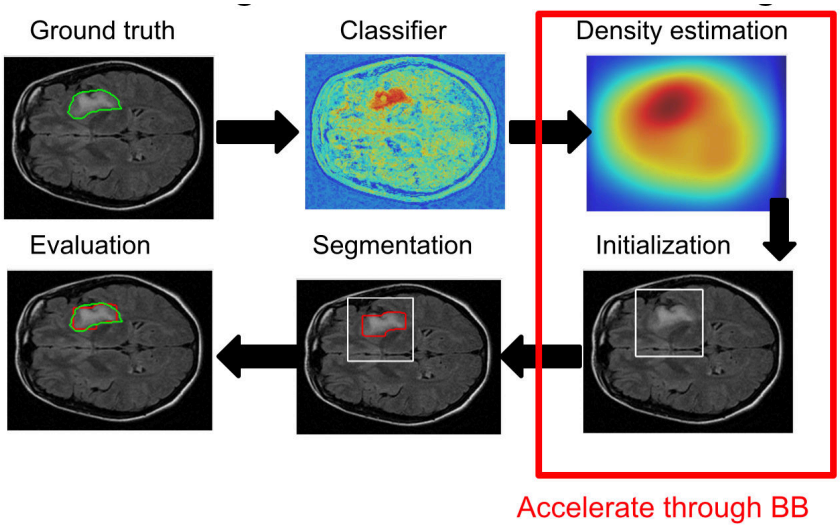


Figure A.2: Our proposed efficient segmentation pipeline; namely, we propose a method to efficiently detect a volume-of-interest through KDE mode estimation.

A.2.1 Problem Formulation

We consider that we are provided with a scoring function that estimates the probability w_i with which a voxel x_i in $\Re^3$ can belong to the considered class (i.e. a tumor vs non-tumor classifier). Namely, we have a mapping:

$$f : \Re^3 \rightarrow [0, 1], x_i \mapsto w_i, \tag{A.1}$$

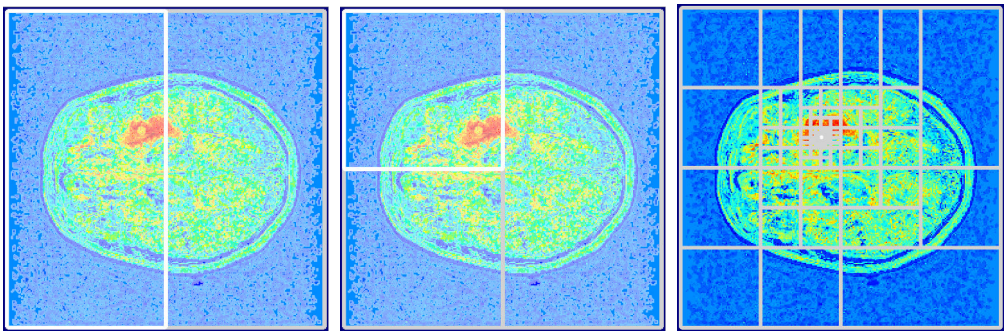
where f encapsulates the feature extraction around x_i and the subsequent formation of the class posterior. In specific, this score can be the output of a soft classifier or a likelihood function on the density distribution of the tumor class.

In order to pool information from multiple voxels and obtain a regularized labeling of the 3D volume, we phrase our problem in terms of a Kernel-based Density Estimator of the form:

$$KDE(x) = \sum_{i=1}^{n} w_i K_h (x - x_i) \tag{A.2}$$

We consider that K_h is a separable decreasing kernel, with the parameter h determining the used amount of smoothing. In the context of our application, we work with the finite-support Epanechnikov kernel [Scott 1992]:

$$K_h(x - x_i) = \begin{cases} 0 \text{ if } \|x - x_i\| > h \\ \frac{3}{4}\left(1 - \left(\frac{\|x - x_i\|}{h}\right)^2\right) \text{ else,} \end{cases} \tag{A.3}$$



First iteration Second iteration Last iteration

Figure A.3: Illustration of our Branch and Bound algorithm ; the prioritized search scheme quickly drives us to the most promising intervals (rectangles) until the first singleton interval is reached. Shown in white is the currently popped interval form the priority queue. Shown in gray are the previously popped intervals and not refined to save computational time.

even though any other separable decreasing kernel could be used, e.g. an uniform or a Gaussian kernel [2]. We also note that in principle we should normalize the w_i elements to have unit sum, but the subsequent tasks are unaffected by this normalization. We address the problem of region detection in terms of mode estimation for the KDE above, namely we set out to find:

$$x^* = \underset{\Re^3}{\operatorname{argmax}}\ S(x) = \underset{\Re^3}{\operatorname{argmax}}\ \sum_{i=1}^{n} w_i K_h\left(x - x_i\right) \tag{A.4}$$

Instead of the iterative procedure employed by Mean-Shift, we now describe how Brand-and-Bound can be used to directly recover the solution of Eq.A.4.

A.2.2 Branch and Bound Algorithm

Branch-and-Bound is an optimization method that searches for the global maximum of a function $S(x)$. To this end, the algorithm employs a recursive subdivision of an interval of solutions X in its prioritized search for the maximum. The priority of an interval is determined by the function's upper bound $\overline{S}$ within it. So, if we consider the maximum value of function S within the interval X, say $S(X) = \max_{x \in X} S(x)$, we bound it with $\overline{S}(X) \geq S(X)$. Moreover, we require $\overline{S}(\{x\}) = S(x)$

At each iteration a candidate domain X is popped from the priority queue, and split into subintervals. A new upper bound for each subinterval is computed and inserted in the priority queue. The bounding function drives BB to the most promising intervals until the first singleton interval, say x, is reached. Since the bound is tight for singletons, we know that the solutions

contained in the remaining intervals of the priority queue will score below x, since the upper bound of their scores is below the returned singleton's score. This guarantees that once a singleton is popped from the priority queue, it will be the global maximum of S.

A.2.3 Bounding the KDE score

Having phrased the general setting of Branch-and-Bound, we now turn to how we can apply it to mode finding for Kernel Density Estimation; the main mathematical construct that we need is a bound on the score of a KDE within an interval of solutions. Namely, we need a function $\overline{S}(X)$ which gives us for an interval X an upper bound to the score of the KDE score within X:

$$\overline{S}(X) \geq \max_{x_j \in X} \sum_{x_i \in X'}^{n} w_i K_h (x_j - x_i) = S(X). \tag{A.5}$$

We call points contained in X' the source locations and points in X the domain locations, with the intuition that the points in X' contribute to a score in X [Gray 2003].

We now decompose the computation of the upper bound in Eq. A.5 into smaller parts by using the partitions $X = \cup_{d \in D} X_d$ and $X' = \cup_{s \in S} X$. Our decomposition is based on the fact that $\max_x f(x) + g(x) \leq \max_x f(x) + \max_x g(x)$. For Eq. A.5 this means that if we separately maximize some of the summands and add them back, this gives us something that will be larger than $S(X)$ (and as such, a valid candidate for $\overline{S}(X)$).

Based on this observation we introduce the quantity:

$$\mu_d^s = \max_{x_j \in X_d} \sum_{x_i \in X_s} w_i K_h (x_j - x_i) \tag{A.6}$$

and upper bound the right-hand side of Eq. A.5 as:

$$S(X) \leq \max_d \sum_s \mu_d^s \tag{A.7}$$

where we have brought the summation over s outside the maximization over x_j. This means that if we can construct separately upper bounds to μ_d^s, we can add them up and obtain a valid upper bound for $S(X)$. This will then be used by Branch-and-Bound to prioritize the search over intervals that contain the density's mode.

In particular, we can upper bound $\mu_X^{X'}$ with $\overline{\mu}_X^{X'}$ as follows:

$$\overline{\mu}_X^{X'} \doteq (\sum_{i \in X'_s} w_i) \max_{i \in X} \max_{j \in X'} K(x_i, x_j) \tag{A.8}$$

The first term in Eq. A.8 can be computed rapidly over large intervals using fine-to-coarse summation.

The computation of the right term in Eq. A.8 can be rapidly implemented if X and X' are rectangular domains, and K is a separable, monotonically nonincreasing kernel. Using simple interval arithmetic [Gray 2003, Kokkinos 2011a, Kokkinos 2012b] can deliver the result of this joint maximization, whose complexity could scale as the product of the cardinalities of X and X' if naively implemented. In particular we can analytically compute the maximal values of $K(x_i, x_j), x_i \in X, x_j \in X'$ in terms of the minimal distances of points in X, X', illustrated as $d_{\min}$ in the right of Fig. A.4 - a detailed description of the term is included in [Kokkinos 2011a, Gray 2003].

Using the quantities in A.8 we can form an upper bound $\overline{S}(X)$ to $S(X)$ as:

$$\overline{S}(X) \doteq \max_X \sum_{X'} \overline{\mu}_X^{X'} \geq S(X) \tag{A.9}$$

The one thing that remains is to control the complexity of the summation over X' in this last equation. In particular, tight bounds require small intervals, but as the intervals become smaller, their number increases. This can become a computational bottleneck if not properly treated.

For this we employ a 'Dual Tree' data structure [Gray 2003, Kokkinos 2011a, Kokkinos 2012b], involving two kd-trees corresponding to the domain and source intervals respectively. This is used in a 'Dual Recursion' algorithm, where the domain intervals X and source intervals X' are simultaneously refined in a coarse-to-fine manner. Starting with two intervals X and X' that correspond to the whole signal domain, at each iteration of the dual recursion these are split, bounded, and pruned if possible.

Pruning is the most crucial aspect of Dual Recursion, and guarantees that the computation remains tractable for fine intervals. The pruning uses a reasoning illustrated on the left side of Fig. A.4: considering that at the current iteration of dual recursion X and X' are partitioned as $X = \{X_A, \ldots, X_D\}$ (domain nodes) and $X' = \{X_1, \ldots, X_6\}$ (source nodes), our goal is to reduce the number of source nodes to be considered at the next iteration as 'supporters' for the children of the current domain nodes. Focusing on a single domain node, X_A, we thus want to find which of $X_1, \ldots, X_6$ to prune; for this we compute not only upper bounds of the contribution of X_i to X_A, but also lower bounds; we denote these as $\overline{\mu}_{X_A}^{X_i}, \underline{\mu}_{X_A}^{X_i}$ respectively. Our reasoning goes as follows: if $\overline{\mu}_{X_A}^{X_i} < \underline{\mu}_{X_A}^{X_j}$, this means that node i (and thus its children) in the best case will contribute something less that what node j (and its children) will contribute in the worst case. We can thus ignore node i when refining the bound for node A. A more extensive discussion is contained in [Kokkinos 2011a, Kokkinos 2012b]. The lower bound $\underline{\mu}_{X_A}^{X_i}$ can be efficiently

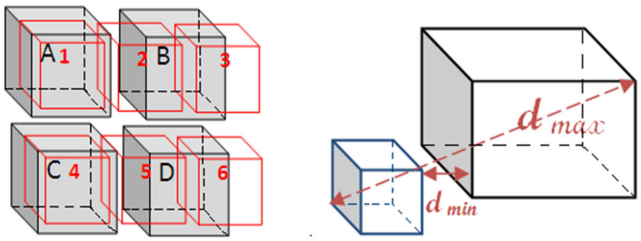


Figure A.4: Left: an example of the rationale behind pruning in Dual Recursion: source nodes are indexes by numbers, domain nodes are indexed by letters; if the upper bound of node 6 contribution to node A does not exceed the lower bound of, say, node 2 contribution to A, node 6 can be pruned. Right: Distance bounds between nodes in dual trees.

computed in terms of the maximal distance between points in X_A, X_i, illustrated as $d_{\max}$ in the right of Fig. A.4.

A.3 3D Brain Tumor Segmentation

Once a region of interest is efficiently selected, we proceed to segmentation in order to delineate the tumor from the normal tissue. Many segmentation methods have been proposed in the literature for tumor segmentation. MRF based-segmentation [Boykov & Kolmogorov 2004] has proved its performance and robustness in many applications. Therefore, we formulate the task of tumor segmentation from the 3D volume of interest as a discrete energy minimization problem. The 3D volume V is viewed as a lattice $\{\vartheta, \varepsilon\}$ where each voxel v_p forms a node in the graph. The MRF energy is written as:

$$E(V) = \sum_{p \in \vartheta} \Theta_p(v_p) + \sum_{(p,q) \in \varepsilon} \Theta_{pq}(v_p, v_q) \tag{A.10}$$

The function f serves as the unary potential energy. In order to improve the classification results, we use a regularization expressed by the binary potential energy. The conventional 4-neighborhood system is extended in 3D so that each voxel has 8 neighbors. We consider, in this work, the Potts model modulated by the contrast of normalized intensities as our regularization function.

This global criterion measures both the total dissimilarity among the two groups and the total similarity inside them. Global minimum of the considered energy is efficiently computed with the graph cut/max flow minimization algorithm[Boykov & Kolmogorov 2004, Kolmogorov & Zabih 2004].

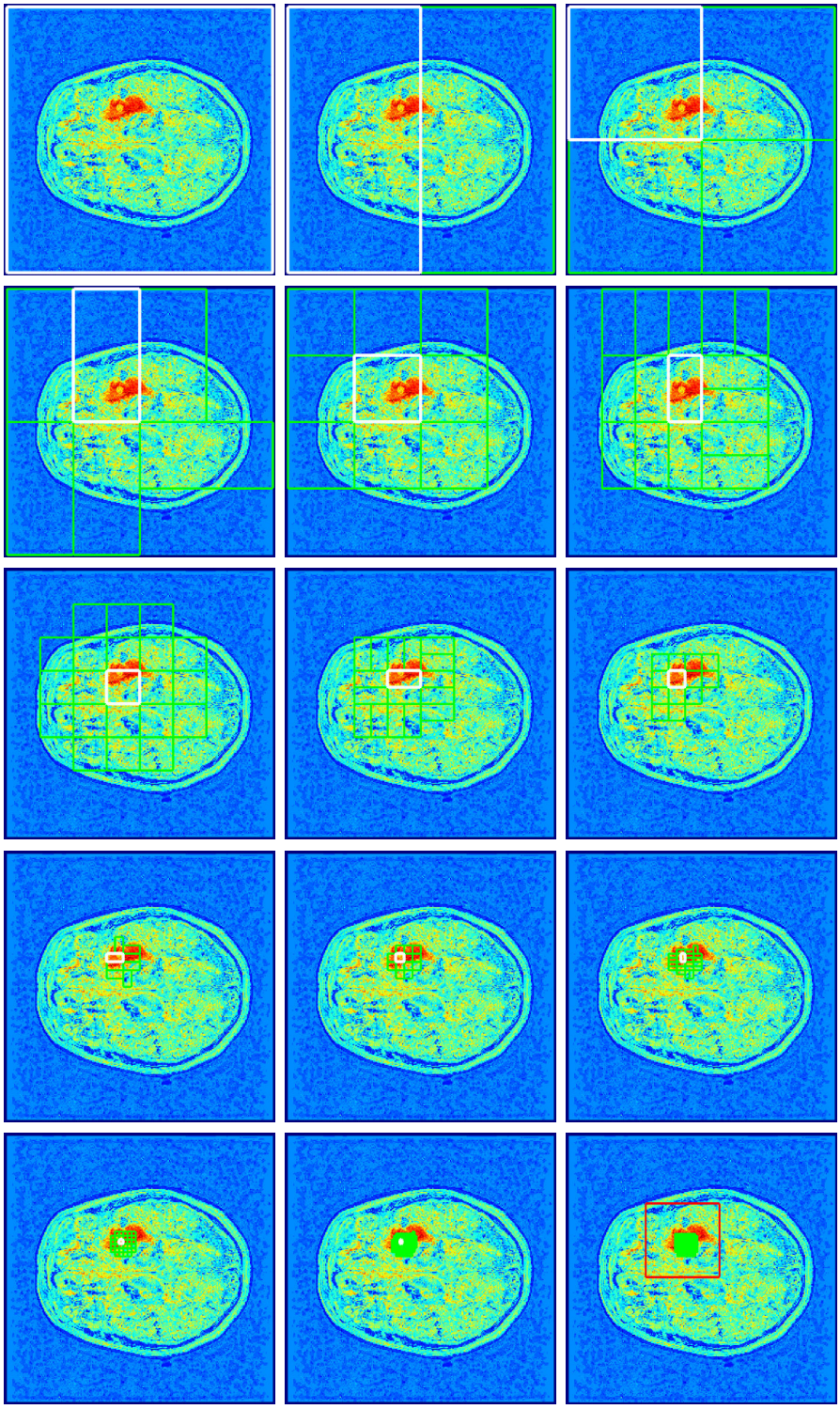


Figure A.5: Popped intervals (white) from the priority queue and their supporters (green) at each iteration; we go simultaneously in a finer level in both candidate rectangles and supporters. At each iteration we prune the supporters that have insignificant contribution to the merit function. This allows us to keep the number of supporters limited, and hence the bounds remain cheaply computable when refining the intervals.

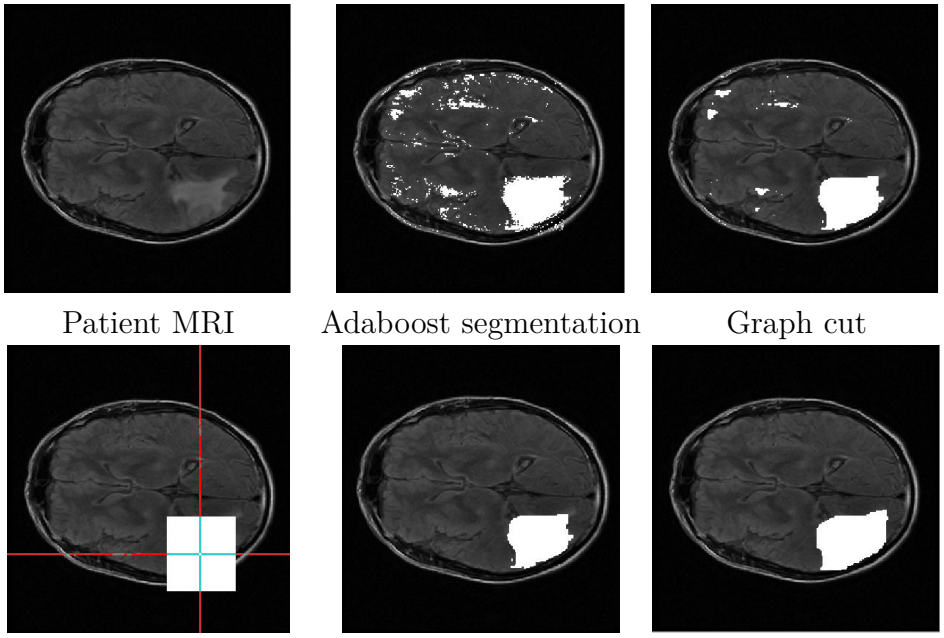


Figure A.6: Comparison between Adaboost segmentation, graph cut segmentation and our method segmentation.

A.4 Experimental Evaluation

image size	profile	our method	Mean-Shift	exhaustive search
256x256x24	detection	2.5s	31s	60s
	overall time	17s	46.5s	75.5s
512x512x33	detection	8s	223s	319s
	overall time	93s	293.5s	389s

Table A.1: Average computational time comparison.

To evaluate our method on a real dataset, we use Adaboost to provide us with the scores of individual voxels. Adaboost is based on the idea that a combination of weak classifiers such as decision trees can create a strong classifier. We use 40 randomly selected images to train the classifier with the following features: normalized intensities, locations (x,y,z), intensities of smoothed image at 3 half octave scales, gradient magnitude, Laplacian of Gaussian features at 3 half octave scales, absolute of Laplacian of Gaussian features at 3 different scales. Our classifier was trained with 50 rounds of boosting and we employed Decision Tress of Depth 3.

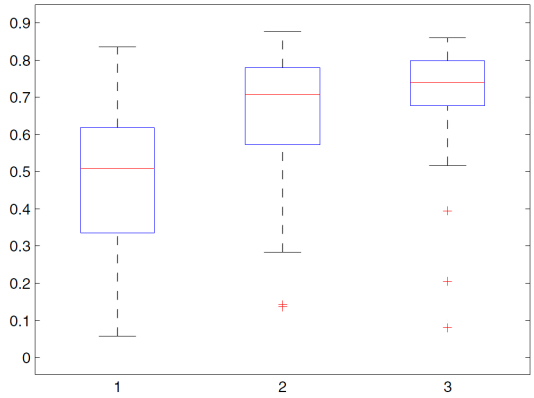


Figure A.7: Boxplots of the Dice values. From left to right: segmentation results with boosting only, boosting and pairwise regularization, boosting, rapid mode estimation and pairwise regularization.

A.4.1 The Dataset

We did our experiments on a dataset composed of 113 patients with low grade gliomas. The patient age ranged from 21 to 65 years, and tumor size between 3.5 and 250 cm^3. The MRI volume size varied from 256x256x24 to 512x512x33. The voxel resolution ranged from 0.4x0.4 to 0.9x0.9 mm^2 in the (x,y) plane and 5.3 to 6.4 mm in the z plane. The 3D images were rigidly aligned using *medInria* [Toussaint *et al.* 2007]. The dataset comes with a manual segmentation of the gliomas tumor done by experts, which is considered as our ground truth data.

A.4.2 Validation Methodology

In order to assess the quality of the segmentation results, we compute the Dice similarity coefficient, which reflects the overlapping rate between the segmented volume and the volume defined by experts. We evaluate the efficiency of our algorithm by comparing the computational time of the detection part with the Mean-Shift procedure and convolving the 3D volume with the kernel. Since the tumor size can achieve 250 cm^3 we convolve with an Epanechnikov kernel whose scale equals nearly the quarter of the volume size. This value matches the maximal size of the ground truth segmented gliomas. We use the most efficient available CPU version of convolution. The used package detects automatically if the kernel is separable and optimizes the convolution computation.

A.4.3 Results

The average Dice computed on our database is 0.73 (cf. Fig. A.7) which is comparable to the results produced by the state of the art methods [Bauer *et al.* 2011, Parisot *et al.* 2012]. we report from [Bauer *et al.* 2011] that the computational cost is between 20 and 120 seconds and the average DICE coefficient is 0.77. Our average computational time is 19 seconds. Mode estimation is a principal ingredient of the proposed method, as the results become less accurate if we only use either Adaboost classifier or graph-cuts (cf. Fig. A.7, Fig. A.4). We compare the computational time between our work, a standard implementation of Mean Shift and an exhaustive search over volume locations after evaluating KDE in all locations cf. Table A.1. We run the algorithms on a 4-core Intel Xeon computer which frequency is 2.67GHz. We use, though, a single core in the computation.

A.5 Relation to relevant techniques

To the best of our knowledge, branch and bound has not been used before for mode estimation of KDEs. It was used in [Kokkinos 2011a] for Object Detection, but with a different score function. We thus expect that our work will also be of interest to researchers working on mode estimation.

A work lying particularly close to ours is the previous work of [Gray 2003], that introduced the Dual Tree data structure, and Dual Recursion. This provides a technique for the efficient computation of a KDE score on all domain points. Similarly, the multipole method [Engheta *et al.* 1985] evaluates a KDE on all candidate locations, and is thus linear in the number of points. The aforementioned methods are excellent for KDE evaluation- but for mode estimation they perform an 'overkill', since they exactly evaluate the score everywhere, while we only want the location of the maximum. Instead our technique searches directly for the maximum location, and effectively ignores less promising locations. This allows us to work in a time that is sublinear (practically logarithmic) in the number of possible locations. This is crucial for 3D medical data, where increasing the resolution by a factor of 2 will result in an 8-fold slowdown for the Multipole/Dual Tree methods, but will only require, in the best case, 3 additional iterations for our method.

A.6 Conclusion

In this paper we have presented a Branch-and-Bound based method for efficient mode estimation in KDE. We used our method for brain tumor detection

and segmentation of 3D MR images. We demonstrate that our method results in a 12-fold speedup over standard Mean-Shift. Our approach is more robust than applying graph cut on the whole volume. The largest part of the computational time is taken by feature computation which can easily be accelerated with graphic processing units. Future directions include evaluating and adapting the proposed approach to the 3D liver tumor tracking in radiation therapy where real time is crucial.

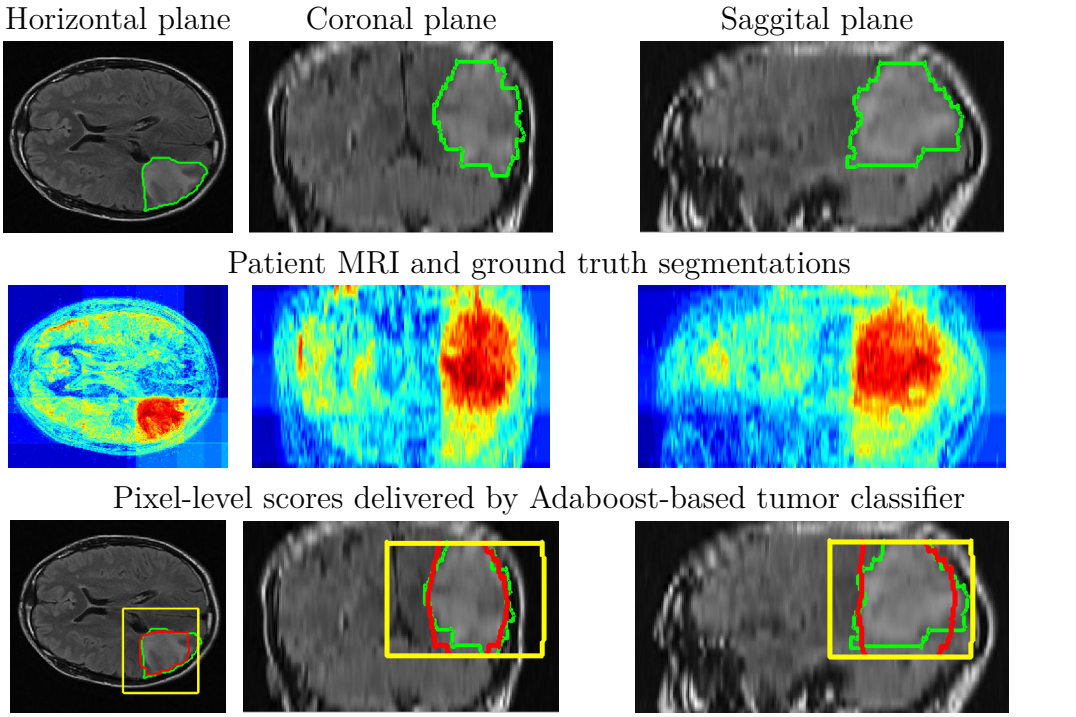


Figure A.8: Illustration of our method on two patient cases: we use ground truth annotations to train pixel-level tumor classifiers using boosting. The pixel-level classifier scores are treated as weights in kernel density estimation KDE; the mode of the resulting KDE can be interpreted as the center of the tumor. We use branch-and-bound to rapidly find the mode of the KDE (yellow box); this is used to initialize the graph-cut segmentation that delivers the contours shown in red.

Appendix B

Hierarchical A^* Algorithm

In this Appendix we describe our implementation of HA^*LD [Felzenszwalb & Mcallester 2007] applied to shortest path search in a trellis graph.

We start by computing the shortest path cost at the coarsest trellis. This level is composed of one node per column describing the whole image domain. Since there is a unique path, we can compute cost so far and cost to go at each node. This allows us to have the heuristic h available for each node in the trellis below -containing 2 nodes per column. We initialize the priority queue with a forward element corresponding to the source node s.

At each iteration, the element with the lowest priority is popped from the $\mathcal{Q}$. Since backward and forward nodes coexist in $\mathcal{Q}$, we need to define which computation is triggered when an element is popped taking into account the current state of the hierarchy.

When a 'forward' element $\{(m, i, \mathbf{x}_i), P_\delta(m, i, \mathbf{x}_i), (m, i-1, \mathbf{x}_{i-1}), f\}$ comes off the queue, we check if the corresponding node to $(i, \mathbf{x}_i)$ in the trellis was not visited before in a cost-so-far computation, we compute its cost-so-far $c(m, i, \mathbf{x}_i)$ and we mark it as visited. Since this cost-so-far is now available, this allows us to update -if any- priorities of elements in $\mathcal{Q}$ that were set to infinity because of unavailability of $c(m, i, \mathbf{x}_i)$ needed in Equation 4.31. At the next trellis column $i+1$ each neighboring node $(m, i+1, \mathbf{x}_{i+1})$ is a potential extension of the shortest path arriving to $(m, i, \mathbf{x}_i)$. In order to decide which is the best extension, we push to $\mathcal{Q}$ elements $\mathcal{E} = \{(m, i+1, \mathbf{x}_{i+1}), P, (m, i, \mathbf{x}_i), f\}$ with a priority expressed by Equation 4.24 and set to infinity if $d(m+1, i+1, \text{pa}(\mathbf{x}_{i+1})$ is not yet computed.

Similarly, when a 'backward' element $\{(m, i, \mathbf{x}_i), P_\delta(m, i, \mathbf{x}_i), (m, i-1, \mathbf{x}_{i-1}), b\}$ comes off the queue, we check if the corresponding node to $(i, \mathbf{x}_i)$ in the trellis was not visited before in a cost-to-go computation, we compute its cost-to-go $d(m, i, \mathbf{x}_i)$ and we mark it as visited. Since this cost-to-go is now available, this allows us to update -if any- priorities of elements in $\mathcal{Q}$ that were set to infinity because of unavailability of $d(m, i, \mathbf{x}_i)$ needed in Equation 4.31. At the next trellis column $i-1$ each neighboring node $(m, i-1, \mathbf{x}_{i-1})$ is a potential extension of the shortest path arriving to $(m, i, \mathbf{x}_i)$ from terminal node t. In order to decide which is the best extension, we push to $\mathcal{Q}$ elements

$\mathcal{E} = \{(m, i-1, \mathbf{x}_{i-1}), P, (m, i, \mathbf{x}_i), b\}$ with a priority expressed by Equation 4.31 and set to infinity if $c(m, i+1, \mathbf{x}_{i+1})$ is not yet computed.

When the terminal node t at a given trellis is reached with a cost $c(m, K, t)$, this triggers the kickoff of a cost-to-go computation starting by terminal node t at the same trellis.

The algorithm stops when the node $(m, i, \mathbf{x}_i)$ popped from $\mathcal{Q}$ belongs to the finest trellis and the last column is K. That is $m = 0$ and $i = K$. We can then backtrack the shortest path in this trellis $\mathcal{G}^0$.

Bibliography

[Abi-Nahed *et al.* 2006] Julien Abi-Nahed, Marie-Pierre Jolly and Guang-Zhong Yang. *Robust Active Shape Models: A Robust, Generic and Simple Automatic Segmentation Tool*. In Medical Image Computing and Computer-Assisted Intervention - MICCAI 2006, 9th International Conference, Copenhagen, Denmark, October 1-6, 2006, Proceedings, Part II, 2006. (Cited on pages 4 and 6.)

[Aji & McEliece 2000] S.M. Aji and R.J. McEliece. *The generalized distributive law*. Information Theory, IEEE Transactions on, 2000. (Cited on page 45.)

[Alomari *et al.* 2011] R. S. Alomari, J. J. Corso and V. Chaudhary. *Labeling of Lumbar Discs using both Pixel- and Object-Level Features with a Two-Level Probabilistic Model*. IEEE TMI, 2011. (Cited on pages x, 14 and 15.)

[Amberg & Vetter 2011] Brian Amberg and Thomas Vetter. *Optimal landmark detection using shape models and branch and bound*. In ICCV, pages 455–462, 2011. (Cited on page 14.)

[Andriluka *et al.* 2012] Mykhaylo Andriluka, Stefan Roth and Bernt Schiele. *Discriminative Appearance Models for Pictorial Structures*. IJCV, vol. 99, no. 3, pages 259–280, 2012. (Cited on pages 9, 12, 13, 14, 41 and 61.)

[Azizpour & Laptev 2012] Hossein Azizpour and Ivan Laptev. *Object Detection Using Strongly-Supervised Deformable Part Models*. In ECCV, 2012. (Cited on pages 41 and 57.)

[Barbu & Gramajo 2014] Adrian Barbu and Gary Gramajo. *Face Detection Using a 3D Model on Face Keypoints*. CoRR, 2014. (Cited on page 13.)

[Bauer *et al.* 2011] Stefan Bauer, Lutz-P. Nolte and Mauricio Reyes. *Fully automatic segmentation of brain tumor images using support vector machine classification in combination with hierarchical conditional random field regularization*. In MICCAI, 2011. (Cited on page 97.)

[Bay *et al.* 2008] Herbert Bay, Andreas Ess, Tinne Tuytelaars and Luc Van Gool. *Speeded-Up Robust Features (SURF)*. Comput. Vis. Image Underst., 2008. (Cited on page 11.)

[Beichel *et al.* 2005] Reinhard Beichel, Horst Bischof, Franz Leberl and Milan Sonka. *Robust active appearance models and their application to medical image analysis*. IEEE Trans. Med. Imaging, 2005. (Cited on page 9.)

[Bellman & Dreyfus 1962] R. Bellman and S.E. Dreyfus. Applied dynamic programming. Princeton University Press, 1962. (Cited on page 61.)

[Bellman 1957] Richard Bellman. Dynamic programming. Princeton University Press, 1957. (Cited on page 61.)

[Bentley 1975] Jon Louis Bentley. *Multidimensional Binary Search Trees Used for Associative Searching*. Commun. ACM, 1975. (Cited on page 69.)

[Bertsekas 1999] D. P. Bertsekas. Nonlinear programming. Athena Scientific, Belmont, MA, 1999. (Cited on pages 44, 46, 48, 49 and 55.)

[Besbes & Paragios 2011] Ahmed Besbes and Nikos Paragios. *Landmark-based segmentation of lungs while handling partial correspondences using sparse graph-based priors*. In ISBI, 2011. (Cited on pages x, 13, 14, 16, 17, 18, 25 and 36.)

[Birkbeck *et al.* 2009] Neil Birkbeck, Dana Cobzas, Martin Jägersand, Albert Murtha and Tibor Kesztyues. *An interactive graph cut method for brain tumor segmentation*. In WACV, 2009. (Cited on page 88.)

[Blanz & Vetter 1999] Volker Blanz and Thomas Vetter. *A Morphable Model for the Synthesis of 3D Faces*. In Annual Conference on Computer Graphics and Interactive Techniques, 1999. (Cited on pages 8 and 17.)

[Bourdev & Malik 2009] Lubomir Bourdev and Jitendra Malik. *Poselets: Body Part Detectors Trained Using 3D Human Pose Annotations*. In ICCV, 2009. (Cited on pages 41 and 57.)

[Boussaid & Kokkinos 2014] Haithem Boussaid and Iasonas Kokkinos. *Fast and Exact: ADMM-Based Discriminative Shape Segmentation with Loopy Part Models*. CVPR, 2014. (Cited on pages xi, 19 and 21.)

[Boussaid *et al.* 2011] Haithem Boussaid, Samuel Kadoury, Iasonas Kokkinos, Jean-Yves Lazennec, Guoyan Zheng and Nikos Paragios. *3D Model-based Reconstruction of the Proximal Femur from Low-dose Biplanar X-Ray Images*. BMVC, 2011. (Cited on pages ix, 4 and 6.)

[Boussaid *et al.* 2013] Haithem Boussaid, Iasonas Kokkinos and Nikos Paragios. *Rapid Mode Estimation for 3D Brain MRI Tumor Segmentation*. EMMCVPR, 2013. (Cited on page 20.)

[Boussaid *et al.* 2014] Haithem Boussaid, Iasonas Kokkinos and Nikos Paragios. *Discriminative Learning of Deformable Contour Models*. ISBI, 2014. (Cited on pages 18 and 25.)

[Boyd *et al.* 2011] Stephen P. Boyd, Neal Parikh, Eric Chu, Borja Peleato and Jonathan Eckstein. *Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers*. Foundations and Trends in Machine Learning, vol. 3, no. 1, pages 1–122, 2011. (Cited on pages 19, 49 and 50.)

[Boykov & Kolmogorov 2004] Yuri Boykov and Vladimir Kolmogorov. *An Experimental Comparison of Min-Cut/Max-Flow Algorithms for Energy Minimization in Vision*. IEEE Trans. Pattern Anal. Mach. Intell., September 2004. (Cited on page 93.)

[Caselles *et al.* 1997] V. Caselles, R. Kimmel and G. Sapiro. *Geodesic Active Contours*. IJCV, vol. 22, no. 1, pages 61–79, 1997. (Cited on pages 3 and 6.)

[Charpiat *et al.* 2007] Guillaume Charpiat, Pierre Maurel, Jean-Philippe Pons, Renaud Keriven and Olivier D. Faugeras. *Generalized Gradients: Priors on Minimization Flows*. IJCV, 2007. (Cited on page 3.)

[Chen *et al.* 2007] Yuanhao Chen, Long Zhu, Chenxi Lin, Alan L. Yuille and HongJiang Zhang. *Rapid Inference on a Novel AND/OR graph for Object Detection, Segmentation and Parsing*. In Proc. NIPS, 2007. (Cited on page 59.)

[Chui & Rangarajan 2003] Haili Chui and Anand Rangarajan. *A New Point Matching Algorithm for Non-rigid Registration*. CVIU, 2003. (Cited on pages 4 and 6.)

[Cohen & Cohen 1993] Laurent D. Cohen and Isaac Cohen. *Finite-Element Methods for Active Contour Models and Balloons for 2-D and 3-D Images*. PAMI, 1993. (Cited on page 3.)

[Comaniciu & Meer 2002] D. Comaniciu and P. Meer. *Mean shift: a robust approach toward feature space analysis*. IEEE Trans. Pattern Anal. Mach. Intell., no. 5, may 2002. (Cited on page 87.)

[Cootes & Taylor 1992] T. F. Cootes and C. J. Taylor. *"Active Shape Models - "Smart Snakes*. In in Proceedings of the British Machine Vision Conference, 1992. (Cited on pages 4, 5 and 6.)

[Cootes *et al.* 1995] Timothy F. Cootes, Christopher J. Taylor, David H. Cooper and Jim Graham. *Active Shape Models-Their Training and Application*. CVIU, vol. 61, pages 38–59, 1995. (Cited on pages 2, 3 and 4.)

[Cootes *et al.* 1998] T.F. Cootes, G. J. Edwards and C.J. Taylor. *Active Appearance Models*. In ECCV, 1998. (Cited on page 2.)

[Cootes *et al.* 2001] Timothy F. Cootes, Gareth J. Edwards and Christopher J. Taylor. *Active Appearance Models*. IEEE Trans. Pattern Anal. Mach. Intell., 2001. (Cited on pages 7 and 8.)

[Cormen *et al.* 2001] Thomas H. Cormen, Clifford Stein, Ronald L. Rivest and Charles E. Leiserson. Introduction to algorithms. 2001. (Cited on page 65.)

[Cremers 2006] D. Cremers. *Dynamical statistical shape priors for level set based tracking*. Trans. PAMI, 2006. (Cited on page 3.)

[D. Zosso & Thiran 2014] X. Bresson D. Zosso and J.P. Thiran. *Fast Geodesic Active Fields for Image Registration based on Splitting and Augmented Lagrangian Approaches*. IEEE Transactions on Image Processing, 2014. (Cited on page 44.)

[Dalal & Triggs 2005] Navneet Dalal and Bill Triggs. *Histograms of Oriented Gradients for Human Detection*. In CVPR, 2005. (Cited on pages 11 and 13.)

[de Bruijne & Nielsen 2004] Marleen de Bruijne and Mads Nielsen. *Shape Particle Filtering for Image Segmentation*. In MICCAI. 2004. (Cited on page 6.)

[de Bruijne *et al.* 2003] Marleen de Bruijne, Bram van Ginneken, Wiro J. Niessen, Marco Loog and Max A. Viergever. Model-based segmentation of abdominal aortic aneurysms in cta images. 2003. (Cited on pages 4 and 6.)

[de La Gorce *et al.* 2011] Martin de La Gorce, David J. Fleet and Nikos Paragios. *Model-Based 3D Hand Pose Estimation from Monocular Video.* IEEE Trans. Pattern Anal. Mach. Intell., 2011. (Cited on page 8.)

[De la Torre & Black 2001] F. De la Torre and M. J. Black. *Robust principal component analysis for computer vision*. In Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on, 2001. (Cited on page 8.)

[Dijkstra 1959] Edsger. W. Dijkstra. *A note on two problems in connexion with graphs.* Numerische Mathematik, 1959. (Cited on page 64.)

[Donner *et al.* 2006] Rene Donner, Michael Reiter, Georg Langs, Philipp Peloschek and Horst Bischof. *Fast Active Appearance Model Search Using Canonical Correlation Analysis.* IEEE Trans. Pattern Anal. Mach. Intell., 2006. (Cited on page 8.)

[Duan *et al.* 2012] Kun Duan, Dhruv Batra and David Crandall. *A Multilayer Composite Model for Human Pose Estimation.* In Proc. British Machine Vision Conference, 2012. (Cited on page 49.)

[Duffau 2004] Capell L. Duffau H. *Preferential brain locations of low-grade gliomas.* Cancer 100, 2004. (Cited on page 87.)

[Durrleman *et al.* 2012] Stanley Durrleman, Marcel Prastawa, JulieR Korenberg, Sarang Joshi, Alain Trouvé and Guido Gerig. *Topology Preserving Atlas Construction from Shape Data without Correspondence using Sparse Parameters.* Med Image Comput Comput Assist Interv, 2012. (Cited on page 1.)

[Ellingsen *et al.* 2010] Lotta Maria Ellingsen, Gouthami Chintalapani, Russell H. Taylor and Jerry L. Prince. *Robust deformable image registration using prior shape information for atlas to patient registration.* Comp. Med. Imag. and Graph., 2010. (Cited on page 1.)

[Engheta *et al.* 1985] Nader Engheta, William D. Murphy, Vladimir Rokhlin and Marius Vassiliou. *The Fast Multipole Method for Electromagnetic Scattering Computation.* IEEE Transactions on Antennas and Propagation, 1985. (Cited on page 97.)

[Felzenszwalb & Huttenlocher 2000] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. *Efficient Matching of Pictorial Structures.* In PROC. IEEE COMPUTER VISION AND PATTERN RECOGNITION CONF., 2000. (Cited on pages ix and 12.)

[Felzenszwalb & Huttenlocher 2004] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. *Distance transforms of sampled functions.* Technical Report Cornell CS, 2004. (Cited on pages iii, vi, 13, 19, 59, 62 and 63.)

[Felzenszwalb & Huttenlocher 2005] P. Felzenszwalb and D. Huttenlocher. *Pictorial Structures for Object Recognition.* IJCV, vol. 61, no. 1, pages 55–79, 2005. (Cited on pages 9, 10, 12, 13, 19, 59, 61 and 62.)

[Felzenszwalb & Mcallester 2007] Pedro F. Felzenszwalb and David Mcallester. *The generalized A* architecture*. Journal of Artificial Intelligence Research, 2007. (Cited on pages iii, iv, v, vi, 19, 59, 60, 66 and 101.)

[Felzenszwalb & Zabih 2011] Pedro F. Felzenszwalb and Ramin Zabih. *Dynamic Programming and Graph Algorithms in Computer Vision*. PAMI, 2011. (Cited on pages 13, 59 and 61.)

[Felzenszwalb *et al.* 2010] Pedro F. Felzenszwalb, Ross B. Girshick and David A. McAllester. *Cascade object detection with deformable part models*. In CVPR, 2010. (Cited on pages ix, 9, 11, 12, 13, 60 and 61.)

[Ferrari *et al.* 2008] Vittorio Ferrari, Manuel J. Marin-Jimenez and Andrew Zisserman. *Progressive search space reduction for human pose estimation*. In Proc. CVPR, 2008. (Cited on page 59.)

[Fischler & Elschlager 1973] M. A. Fischler and R. A. Elschlager. *The Representation and Matching of Pictorial Structures*. IEEE Trans. Comput., 1973. (Cited on pages ix, 9 and 10.)

[Freeman & Adelson 1991] William T. Freeman and Edward H. Adelson. *The Design and Use of Steerable Filters*. PAMI, 1991. (Cited on pages 6, 36 and 37.)

[Freund & Schapire 1997] Yoav Freund and Robert E. Schapire. *A Decision-theoretic Generalization of On-line Learning and an Application to Boosting*. J. Comput. Syst. Sci., 1997. (Cited on page 6.)

[Fulkerson *et al.* 2008] Brian Fulkerson, Andrea Vedaldi and Stefano Soatto. *Localizing Objects with Smart Dictionaries*. In ECCV, 2008. (Cited on pages 11, 19, 36 and 37.)

[Geiger *et al.* 1995] Davi Geiger, Alok Gupta, Luiz A. Costa and John Vlontzos. *Dynamic Programming for Detecting, Tracking, and Matching Deformable Contours*. PAMI, 1995. (Cited on pages 3 and 59.)

[Goldstein & Osher 2009] Tom Goldstein and Stanley Osher. *The Split Bregman Method for L1-Regularized Problems*. SIAM J. Imaging Sciences, 2009. (Cited on page 44.)

[Goulden 2008] C. H Goulden. Methods of statistical analysis. 2008. (Cited on page 41.)

[Gray 2003] Alexander G. Gray. *Nonparametric density estimation: toward computational tractability.* In SIAM International Conference on Data Mining., 2003. (Cited on pages 85, 91, 92 and 97.)

[Gu 2012] Steve Gu. *Extended Subwindow Search and Pictorial Structures.* PhD thesis, DUKE University, 2012. (Cited on page 83.)

[Hart *et al.* 1968] P. E. Hart, N. J. Nilsson and B. Raphael. *A formal basis for the heuristic determination of minimum cost paths.* IEEE Transactions on Systems, Science, and Cybernetics, 1968. (Cited on page 65.)

[Hart *et al.* 1972] Peter E. Hart, Nils J. Nilsson and Bertram Raphael. *Correction to "A Formal Basis for the Heuristic Determination of Minimum Cost Paths".* SIGART Bull., 1972. (Cited on page 65.)

[Hazan & Urtasun 2010] Tamir Hazan and Raquel Urtasun. *A Primal-Dual Message-Passing Algorithm for Approximated Large Scale Structured Prediction.* In Advances in Neural Information Processing Systems 23. 2010. (Cited on page 83.)

[Heimann & Meinzer 2009] Tobias Heimann and Hans-Peter P. Meinzer. *Statistical shape models for 3D medical image segmentation: a review.* Medical image analysis, 2009. (Cited on page 1.)

[Held & Karp 1971] Michael Held and RichardM. Karp. *The traveling-salesman problem and minimum spanning trees: Part II.* Mathematical Programming, 1971. (Cited on page 66.)

[Isard & Blake 1998] Michael Isard and Andrew Blake. *CONDENSATION - conditional density propagation for visual tracking.* IJCV, 1998. (Cited on page 6.)

[Ishida *et al.* 2005] Takayuki Ishida, Shigehiko Katsuragawa, Koichi Chida, Heber MacMahon and Kunio Doi. *Computer-aided diagnosis for detection of cardiomegaly in digital chest radiographs.* Proc. SPIE, 2005. (Cited on page 1.)

[Jiao *et al.* 2003] Feng Jiao, Stan Li, Heung-Yeung Shum and D. Schuurmans. *Face alignment using statistical models and wavelet features.* In CVPR, 2003. (Cited on page 6.)

[Joachims *et al.* 2009] Thorsten Joachims, Thomas Finley and Chun-Nam John Yu. *Cutting-plane training of structural SVMs.* Machine Learning, 2009. (Cited on pages 18, 28 and 30.)

[Kass *et al.* 1987] M. Kass, A. Witkin and D. Terzopoulos. *Snakes: Active Contour Models*. In ICCV, 1987. (Cited on page 2.)

[Ke & Sukthankar 2004] Yan Ke and Rahul Sukthankar. *PCA-SIFT: A More Distinctive Representation for Local Image Descriptors*. In CVPR, 2004. (Cited on page 11.)

[Kimia *et al.* 2003] Benjamin B. Kimia, Ilana Frankel and Ana-Maria Popescu. *Euler Spiral for Shape Completion*. International Journal of Computer Vision, 2003. (Cited on page 3.)

[Kokkinos & Yuille 2008] Iasonas Kokkinos and Alan Yuille. *Scale invariance without scale selection*. CVPR, 2008. (Cited on pages 11 and 83.)

[Kokkinos 2011a] Iasonas Kokkinos. *Rapid Deformable Object Detection using Dual-Tree Branch-and-Bound*. In J. Shawe-Taylor, R.S. Zemel, P. Bartlett, F.C.N. Pereira and K.Q. Weinberger, editeurs, NIPS. 2011. (Cited on pages iii, vi, xiii, 66, 71, 85, 87, 92 and 97.)

[Kokkinos 2011b] Iasonas Kokkinos. *Rapid Deformable Object Detection using Dual-Tree Branch-and-Bound*. NIPS, 2011. (Cited on pages 59, 60, 71 and 74.)

[Kokkinos 2012a] I. Kokkinos. *Bounding Part Scores for Rapid Detection with Deformable Part Models*. In 2nd Parts and Attributes Workshop, in conjunction with ECCV 2012, 2012. (Cited on page 87.)

[Kokkinos 2012b] Iasonas Kokkinos. *Rapid Deformable Object Detection using Bounding-based Techniques*. Rapport technique RR-7940, INRIA, 2012. (Cited on pages 87 and 92.)

[Kolmogorov & Zabih 2004] Vladimir Kolmogorov and Ramin Zabih. *What energy functions can be minimized via graph cuts*. PAMI, 2004. (Cited on page 93.)

[Kolmogorov 2006] Vladimir Kolmogorov. *Convergent Tree-Reweighted Message Passing for Energy Minimization*. IEEE Trans. Pattern Anal. Mach. Intell., 2006. (Cited on page 16.)

[Komodakis *et al.* 2007] Nikos Komodakis, Nikos Paragios and Georgios Tziritas. *MRF optimization via dual decomposition: Message-passing revisited*. In ICCV, 2007. (Cited on pages 44 and 49.)

[Komodakis *et al.* 2011] Nikos Komodakis, Nikos Paragios and Georgios Tziritas. *MRF Energy Minimization and Beyond via Dual Decomposition.* IEEE Trans. Pattern Anal. Mach. Intell., vol. 33, no. 3, pages 531–552, 2011. (Cited on page 49.)

[Komodakis 2011] N. Komodakis. *Efficient Training for Pairwise or Higher Order CRFs via Dual Decomposition.* In CVPR, 2011. (Cited on page 83.)

[Kumar & Torr 2006] M Pawan Kumar and Philip H.S. Torr. *Fast Memory-Efficient Generalized Belief Propagation.* the Proceedings of the Ninth European Conference on Computer Vision 2006, 2006. (Cited on page 59.)

[Kumar *et al.* 2011] M.P. Kumar, H. Turki, D. Preston and D. Koller. *Learning Specific-Class Segmentation from Diverse Data.* In Proceedings of the International Conference on Computer Vision (ICCV), 2011. (Cited on page 13.)

[Langs *et al.* 2006] Georg Langs, Philipp Peloschek, Rene Donner, Michael Reiter and Horst Bischof. *Active Feature Models.* In ICPR, 2006. (Cited on pages 4 and 6.)

[Leventon *et al.* 2000] M. Leventon, O. Faugeras and E. Grimson. *Statistical Shape Influence in Geodesic Active Contours.* In CVPR, 2000. (Cited on page 3.)

[Lewis *et al.* 2007] R. Lewis, A. Shepherd and V. Torczon. *Implementing generating set search methods for linearly constrained minimization.* SIAM J. Sci. Comput., vol. 6, pages 2507–2530, 2007. (Cited on page 6.)

[Li *et al.* 2004] Shuyu Li, Litao Zhu and Tianzi Jiang. *Active Shape Model Segmentation Using Local Edge Structures and AdaBoost.* In MIAR, 2004. (Cited on pages 4 and 6.)

[Lowe 2004] David G. Lowe. *Distinctive Image Features from Scale-Invariant Keypoints.* Int. J. Comput. Vision, 2004. (Cited on pages 11 and 36.)

[Malladi *et al.* 1995] Ravi Malladi, James A. Sethian and Baba C. Vemuri. *Shape Modeling with Front Propagation: A Level Set Approach.* PAMI, 1995. (Cited on page 3.)

[Martins *et al.* 2011a] Andre Martins, Mario Figueiredo, Pedro Aguiar, Noah Smith and Eric Xing. *An Augmented Lagrangian Approach to Constrained MAP Inference.* In ICML, 2011. (Cited on pages 19 and 44.)

[Martins *et al.* 2011b] André F. T. Martins, Noah A. Smith, Pedro M. Q. Aguiar and Mário A. T. Figueiredo. *Dual decomposition with many overlapping components*. In EMNLP, 2011. (Cited on pages 49, 51 and 55.)

[Matthews & Baker 2004] Iain Matthews and Simon Baker. *Active Appearance Models Revisited*. Int. J. Comput. Vision, 2004. (Cited on pages 7 and 8.)

[Mayer & Greenspan 2009] Arnaldo Mayer and Hayit Greenspan. *An Adaptive Mean-Shift Framework for MRI Brain Segmentation*. IEEE Transactions on Medical Imaging, 2009. (Cited on page 87.)

[Mikolajczyk & Schmid 2005] Krystian Mikolajczyk and Cordelia Schmid. *A Performance Evaluation of Local Descriptors*. IEEE Trans. Pattern Anal. Mach. Intell., 2005. (Cited on page 11.)

[Mittal *et al.* 2012] Arpit Mittal, Matthew B. Blaschko, Andrew Zisserman and Philip H. S. Torr. *Taxonomic Multi-class Prediction and Person Layout Using Efficient Structured Ranking*. In ECCV, 2012. (Cited on page 28.)

[Nakamori *et al.* 1991] N. Nakamori, K. Doi, H. Macmahon, Y. Sasaki and S. Montner. *Effect of Heart-Size Parameters Computed from Digital Chest Radiographs on Detection of Cardiomegaly: Potential Usefulness for Computer-Aided Diagnosis*. Investigative Radiology, 1991. (Cited on page 1.)

[Nowozin & Lampert 2011] Sebastian Nowozin and Christoph H. Lampert. *Structured Learning and Prediction in Computer Vision*. Foundations and Trends in Computer Graphics and Vision, 2011. (Cited on pages ix, 12, 18 and 28.)

[Paragios 2003] Nikos Paragios. *A Level Set Approach for Shape-driven Segmentation and Tracking of the Left Ventricle*. IEEE Trans. Med. Imaging, 2003. (Cited on page 1.)

[Parisot *et al.* 2012] Sarah Parisot, Hugues Duffau, Stéphane Chemouny and Nikos Paragios. *Graph-based detection, segmentation & characterization of brain tumors*. In CVPR, 2012. (Cited on pages 87 and 97.)

[Pearl 1984] Judea Pearl. Heuristics: Intelligent search strategies for computer problem solving. 1984. (Cited on page 66.)

[Polyak 1967] B. T. Polyak. *A General Method of Solving Extremum Problems*. 1967. (Cited on page 49.)

[Potesil *et al.* 2010] Vaclav Potesil, Timor Kadir, Gunther Platsch and Mike Brady. *Improved Anatomical Landmark Localization in Medical Images Using Dense Matching of Graphical Models*. In BMVC, 2010. (Cited on pages x, 14, 25 and 36.)

[Potesil *et al.* 2011] Vaclav Potesil, Timor Kadir, Gunther Platsch and Michael Brady. *Personalization of Pictorial Structures for Anatomical Landmark Localization*. In IPMI, 2011. (Cited on pages x, 14 and 36.)

[Potesil *et al.* 2014] V. Potesil, T. Kadir and M. Brady. *Learning New Parts for Landmark Localization in Whole-Body CT Scans*. Medical Imaging, IEEE Transactions on, 2014. (Cited on page 14.)

[Prieditis 1991] Armand Prieditis. *Machine Discovery of Effective Admissible Heuristics*. In IJCAI, 1991. (Cited on page 66.)

[Prima *et al.* 2008] Daniel Garcia-Lorenzoand Sylvain Prima, D. Louis Collins, Douglas L. Arnold, Sean P. Morrissey and Christian Barillot. *Combining Robust Expectation Maximization and Mean Shift algorithms for Multiple Sclerosis Brain Segmentation*. MICCAI workshop on Medical Image Analysis on Multiple Sclerosis., 2008. (Cited on page 87.)

[Prince 2012] S.J.D. Prince. Computer Vision: Models Learning and Inference. Cambridge University Press, 2012. (Cited on pages ix and 7.)

[Raphael 2001] Christopher Raphael. *Coarse-to-Fine Dynamic Programming*. PAMI, vol. 23, pages 1379–1390, 2001. (Cited on page 60.)

[Rochery *et al.* 2006] Marie Rochery, Ian H. Jermyn and Josiane Zerubia. *Higher Order Active Contours*. IJCV, 2006. (Cited on page 3.)

[Rousson & Paragios 2002] M. Rousson and N. Paragios. *Shape Priors for Level Set Representations*. In ECCV, 2002. (Cited on pages 3 and 6.)

[Salzmann 2013] Mathieu Salzmann. *Continuous Inference in Graphical Models with Polynomial Energies*. In CVPR, 2013. (Cited on page 44.)

[Sapp *et al.* 2010] Benjamin Sapp, Alexander Toshev and Ben Taskar. *Cascaded Models for Articulated Pose Estimation*. In ECCV, 2010. (Cited on pages 9, 12, 13, 14, 41, 59 and 61.)

[Sapp *et al.* 2011a] Benjamin Sapp, David Weiss and Ben Taskar. *Parsing human motion with stretchable models.* In CVPR, 2011. (Cited on pages 9, 12, 13, 14, 41, 44 and 61.)

[Sapp *et al.* 2011b] Benjamin Sapp, David Weiss and Ben Taskar. *Parsing human motion with stretchable models.* In CVPR, 2011. (Cited on pages 28 and 41.)

[Saragih & Göcke 2007] Jason Saragih and Roland Göcke. *A Nonlinear Discriminative Approach to AAM Fitting.* In ICCV, pages 1–8, 2007. (Cited on page 14.)

[Schmidt *et al.* 2007] Stefan Schmidt, Jorg H. Kappes, Martin Bergtholdt, Vladimir Pekar, Sebastian P. M. Dries, Daniel Bystrov and Christoph Schnorr. *Spine Detection and Labeling Using a Parts-Based Graphical Model.* In IPMI, 2007. (Cited on pages x, 14, 15 and 36.)

[Scott 1992] D.W. Scott. Multivariate density estimation: theory, practice, and visualization. Wiley series in probability and mathematical statistics: Applied probability and statistics. Wiley., 1992. (Cited on page 89.)

[scr] *SCR database.* http://www.isi.uu.nl/Research/Databases/SCR /results/browser.php. Accessed: 2013-10-01. (Cited on page 40.)

[Seghers *et al.* 2007] Dieter Seghers, Dirk Loeckx, Frederik Maes, Dirk Vandermeulen and Paul Suetens. *Minimal Shape and Intensity Cost Path Segmentation.* IEEE Trans. Med. Imaging, 2007. (Cited on pages iii, v, x, xi, xii, xvii, 1, 15, 16, 17, 18, 24, 25, 32, 33, 37, 39, 40, 41 and 42.)

[Sheikh *et al.* 2007] Yaser Ajmal Sheikh, E.Khan and Takeo Kanade. *Mode-seeking by Medoidshifts.* In ICCV., numéro 141., October 2007. (Cited on page 87.)

[Shiraishi *et al.* 2000] J. Shiraishi, S. Katsuragawa, T. Matsumoto J. Ikezoe, T. Kobayashi, K. Komatsu, M. Matsui, H. Fujita, Y. Kodera and K. Doi. *Development of a digital image database for chest radiographs with and without a lung nodule: Receiver operating characteristic analysis of radiologists detection of pulmonary nodules.* Am. J. Roentgenology, 2000. (Cited on pages iii, v, xvii, 18, 31, 33, 41 and 76.)

[Sontag *et al.* 2011] David Sontag, Amir Globerson and Tommi Jaakkola. *Introduction to Dual Decomposition for Inference.* In Suvrit Sra, Sebastian Nowozin and Stephen J. Wright, editeurs, Optimization for Machine Learning. MIT Press, 2011. (Cited on page 44.)

[Stegmann *et al.* 2001] M. B. Stegmann, R. Fisker and B. K. Ersboll. *Extending and applying active appearance models for automated, high precision segmentation in different image modalities.* 2001. (Cited on pages 8 and 9.)

[Stegmann 2002] M. B. Stegmann. *Analysis and Segmentation of Face Images using Point Annotations and Linear Subspace Techniques.* Technical Report, 2002. (Cited on pages ix and 7.)

[Sundaramoorthi *et al.* 2008] Ganesh Sundaramoorthi, Anthony J. Yezzi and Andrea Mennucci. *Coarse-to-Fine Segmentation and Tracking Using Sobolev Active Contours.* PAMI, 2008. (Cited on page 3.)

[Toews & Wells III 2012] M. Toews and W.M. Wells III. *Efficient and Robust Model-to-Image Alignment using 3D Scale-Invariant Features.* Medical Image Analysis, 2012. (Cited on page 36.)

[Tola *et al.* 2008] E. Tola, V. Lepetit and P. Fua. *A Fast Local Descriptor for Dense Matching.* In Proc. CVPR, 2008. (Cited on pages xi, 11, 19, 36 and 37.)

[Torre & Black 2003] Fernando De La Torre and Michael J. Black. *A Framework for Robust Subspace Learning.* International Journal of Computer Vision, 2003. (Cited on page 7.)

[Toussaint *et al.* 2007] Nicolas Toussaint, Jean christophe Souplet and Pierre Fillard. *Medinria: medical image navigation and research tool by INRIA.* In Proceedings of MICCAI Workshop on Interaction in Medical Image Analysis and Visualization, 2007. (Cited on page 96.)

[van Assen *et al.* 2003] Hans C. van Assen, Mikhail G. Danilouchkine, Faiza Behloul, Hildo J. Lamb, Rob J. van der Geest, Johan H. C. Reiber and Boudewijn P. F. Lelieveldt. *Cardiac LV Segmentation Using a 3D Active Shape Model Driven by Fuzzy Inference.* In MICCAI (1), 2003. (Cited on pages 4 and 6.)

[van Ginneken *et al.* 2002] Bram van Ginneken, Alejandro F. Frangi, Ro F. Frangi, Joes J. Staal, Bart M. Ter Haar Romeny and Max A. Viergever. *Active Shape Model Segmentation with Optimal Features.* IEEE Transactions on Medical Imaging, 2002. (Cited on pages 1, 4 and 6.)

[van Ginneken *et al.* 2006] B. van Ginneken, M.B. Stegmann and M. Loog. *Segmentation of anatomical structures in chest radiographs using supervised methods: a comparative study on a public database.* Medical

Image Analysis, 2006. (Cited on pages iii, v, xi, xii, xvii, 8, 9, 31, 32, 33, 35, 39, 40, 41, 42 and 76.)

[Varshney *et al.* 2009] K. Varshney, N. Paragios, A. Kulski, R. Raymond, P. Hernigou and A. Rahmouni. *Post-Arthroplasty Examination Using X-Ray Images.* IEEE Transactions on Medical Imaging, vol. 28, pages 469–474, 2009. (Cited on page 6.)

[Vedaldi & Soatto 2008] A. Vedaldi and S. Soatto. *Quick Shift and Kernel Methods for Mode Seeking.* In ECCV., 2008. (Cited on page 87.)

[Vedaldi *et al.* 2014] Andrea Vedaldi, S. Mahendran, S. Tsogkas, S. Maji, B. Girshick, J. Kannala, E. Rahtu, I. Kokkinos, M. B. Blaschko, D. Weiss, B. Taskar, K. Simonyan, N. Saphra and S. Mohamed. *Understanding Objects in Detail with Fine-grained Attributes.* In CVPR, 2014. (Cited on pages 41 and 57.)

[Walsum *et al.* 2008] T.W. van Walsum, M. Schaap, C.T. Metz, A. van der Giessen and W.J. Niessen. *Averaging centerlines: Mean shift on paths.* MICCAI, 2008. (Cited on page 87.)

[Wang *et al.* 2007] Ping Wang, Dongryeol Lee, Alexander G. Gray and James M. Rehg. *Fast Mean Shift with Accurate and Stable Convergence.* Journal of Machine Learning Research - Proceedings Track, 2007. (Cited on page 87.)

[Wang *et al.* 2010a] Chaohui Wang, Olivier Teboul, Fabrice Michel, Salma Essafi and Nikos Paragios. *3D Knowledge-Based Segmentation Using Pose-Invariant Higher-Order Graphs.* In Miccai, 2010. (Cited on page 36.)

[Wang *et al.* 2010b] Chaohui Wang, Olivier Teboul, Fabrice Michel, Salma Essafi and Nikos Paragios. *3D Knowledge-Based Segmentation Using Pose-Invariant Higher-Order Graphs.* In MICCAI, 2010. (Cited on page 44.)

[Wang *et al.* 2011] Chaohui Wang, Haithem Boussaid, Loïc Simon, Jean-Yves Lazennec and Nikos Paragios. *Pose-Invariant 3D Proximal Femur Estimation through Bi-planar Image Segmentation with Hierarchical Higher-Order Graph-Based Priors.* In MICCAI, 2011. (Cited on page 44.)

[Xiang *et al.* 2012] Bo Xiang, Chaohui Wang, Jean-François Deux, Alain Rahmouni and Nikos Paragios. *3D cardiac segmentation with pose-invariant higher-order MRFs.* In ISBI, 2012. (Cited on page 44.)

[Xiang *et al.* 2013] Bo Xiang, Jean-François Deux, Alain Rahmouni and Nikos Paragios. *Joint Model-Pixel Segmentation with Pose-Invariant Deformable Graph-Priors*. In MICCAI, 2013. (Cited on pages xiv and 84.)

[Yang *et al.* 2003] Changjiang Yang, Ramani Duraiswami, Nail A. Gumerov and Larry S. Davis. *Improved Fast Gauss Transform and Efficient Kernel Density Estimation*. In ICCV, 2003. (Cited on page 87.)

[Yin *et al.* 2008] Wotao Yin, Stanley Osher, Donald Goldfarb and Jérôme Darbon. *Bregman Iterative Algorithms for L1-Minimization with Applications to Compressed Sensing*. SIAM J. Imaging Sciences, 2008. (Cited on page 44.)

[Yuille *et al.* 1992] A. L. Yuille, P. W. Hallinan and D. S. Cohen. *Feature Extraction from Faces Using Deformable Templates*. IJCV, vol. 8, no. 2, pages 99–111, 1992. (Cited on page 2.)

[Zhu & Ramanan 2012] Xiangxin Zhu and Deva Ramanan. *Face detection, pose estimation, and landmark localization in the wild*. In CVPR, 2012. (Cited on pages 9, 10, 12, 13 and 61.)