CHAPTER

Efficient Medical Image Parsing

3

Florin C. Ghesu[*,†], **Bogdan Georgescu**[*], **Joachim Hornegger**[†]
Siemens Medical Solutions USA, Inc., Princeton, NJ, United States[*] *Friedrich-Alexander University Erlangen–Nürnberg, Erlangen, Germany*[†]

CHAPTER OUTLINE

3.1 Introduction 55
3.2 Background and Motivation 57
3.2.1 Object Localization and Segmentation: Challenges 58
3.3 Methodology 58
3.3.1 Problem Formulation 58
3.3.2 Sparse Adaptive Deep Neural Networks 59
3.3.3 Marginal Space Deep Learning 61
3.3.3.1 Cascaded Negative Filtering 63
3.3.3.2 Nonrigid Deformation Estimation 63
3.3.4 An Artificial Agent for Image Parsing 64
3.3.4.1 Cognitive Modeling Using Deep Reinforcement Learning 65
3.3.4.2 Learning to Scan for Anatomical Objects 67
3.4 Experiments 71
3.4.1 Anatomy Detection and Segmentation in 3D 71
3.4.1.1 Dataset 71
3.4.1.2 Model Selection and Training Parameters 71
3.4.1.3 Evaluation 71
3.4.1.4 Additional Experiments and Discussion 73
3.4.2 Landmark Detection in 2D and 3D 74
3.4.2.1 Datasets 74
3.4.2.2 Evaluation: Learning to Parse 75
3.5 Conclusion 77
Disclaimer 78
References 78

3.1 INTRODUCTION

Powerful data representations drive the performance of many machine learning algorithms [1]. Designing features that generate such representations to effectively

Deep Learning for Medical Image Analysis. DOI: 10.1016/B978-0-12-810408-8.00005-5

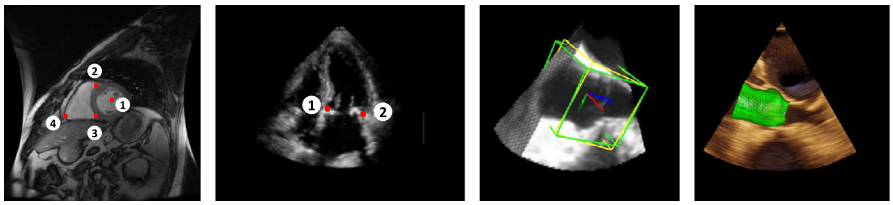


FIGURE 3.1

Example images displaying parsed anatomical objects. Figures A and B show anatomical landmarks in cardiac magnetic resonance and ultrasound images, while C and D show the bounding box and boundary mesh for the aortic root in 3D ultrasound.

capture the information encoded in the given data is a particularly difficult task [2–5]. In practice, this requires complex data preprocessing pipelines that do not generalize well between different image modalities or learning problems. The reason for that is that most of these systems are manually engineered for specific applications and rely exclusively on human ingenuity to disentangle and understand prior information hidden in the data in order to design the required features and discover intrinsic information about the given task [1].

In the context of volumetric image parsing, machine learning is used for anatomy detection and organ segmentation [5,6] (see Fig. 3.1). Here, the task of feature engineering becomes increasingly complex since the feature extraction is typically performed under challenging transformations such as arbitrary orientations or scales. Moreover, for robust parameter estimation, scanning the parameter space exhaustively is not feasible given the exponential increase in the size of the space with respect to the space dimensionality, i.e. the number of considered transformation parameters.

We overcome all these challenges by proposing *Marginal Space Deep Learning* (MSDL), a feature-learning-based framework to support the efficient parsing of arbitrary anatomical structures. For this we formulate a two-step approach using deep learning (DL) as a powerful solution for joint feature learning and task learning in each step: object localization and boundary estimation. The framework relies on the computational advantages of scanning hierarchical parametric spaces through the mechanism of marginal space learning [5]. In the case of a restricted affine transformation, we estimate first the location (3D space), then the location and orientation (6D space), and finally the complete transformation including the anisotropic scaling (9D space). Given the rigid transformation of the object we propose a deep-learning-based active shape model (ASM) [7] to guide the shape deformation starting from the mean shape. For a defined object parametrization, the system learns classifiers in marginal parameter spaces thereby capturing the appearance of the object under specific transformations. We present two principled solutions for this, one based on exhaustive hypotheses scanning using advanced cascade filtering and sparse feature

sampling, and the second following a more intelligent search scheme, learning optimal trajectories in parameter space that lead to the target location describing the actual object transformation.

The first part of this chapter is dedicated to the scanning-based solution which we published in [8,9]. As exhaustive scanning comes at a high computational cost, the current applications of state-of-the-art DL architectures are focused on pixel(voxel)-wise classification in 2D or 2.5D data, with no generic extension supporting the parsing of large volumetric images. Capturing the complex appearance of 3D structures and ensuring the efficient scanning of high-dimensional parameter spaces are not straightforward given that the size of the parameter space increases exponentially with respect to the number of transformation parameters. To account for this we propose cascaded *sparse adaptive deep neural networks* (SADNN) to learn parametrized representations from 3D medical image modalities and enable the efficient scanning of large parameter spaces. We propose to use sparsity as a means to simplify the network and adaptively focus its sampling pattern on context-rich parts of the input by explicitly discarding neural connections. Note that this is different from typical weight-dropping regularization approaches like dropout [10]. We validate this solution on the problem of detecting and segmenting the aortic heart valve using a large 3D ultrasound dataset.

In the second part of the chapter, we present an alternative solution first introduced in [11], which unifies the learning of the appearance model and the object search to a behavioral problem for an artificial agent. Inspired by the fundamental work of Mnih et al. [12], we leverage state-of-the-art representation learning techniques through deep learning [13,14] and powerful solutions for generic behavior learning through reinforcement learning [15,16,12] to create a model encapsulating a cognitive-like learning process covering both the appearance of the object and search strategy. Initial results on the task of landmark detection prove that this type of approach is very promising in terms of accuracy, robustness, and particularly speed by avoiding typical inefficient scanning routines.

3.2 BACKGROUND AND MOTIVATION

Medical image parsing subsumes the robust recognition, detection, and segmentation of objects which proves to be a particularly difficult task for arbitrary 3D anatomical structures considering the variance in location, the nonrigid nature of the shape, as well as the differences in anatomy among different cases [5]. Many solutions focus mainly on the segmentation task, for example, Active Shape Models [17,7] and deformable models [18,19]. Nonetheless, in order to achieve an accurate automatic segmentation of arbitrary anatomical structures, a robust and efficient solution is required for localizing the object of interest.

3.2.1 OBJECT LOCALIZATION AND SEGMENTATION: CHALLENGES

Initially introduced in the 2D context [4,3], machine learning can be used for the efficient and robust localization of objects. In this context object localization is formulated as a classification problem over patches of image intensities extracted from transformed boxes defined in a parametric space $\mathcal{U}$. Typically a classifier is learned in this space using discrete positive and negative hypotheses $h \in \mathcal{U}$. During testing the classifier is applied to exhaustively scan the parameter space, an effort which grows exponentially with the dimensionality of the space. Extending the logic to 3D is however not straightforward. Let us start from the example of a restricted affine transformation. This type of rigid transformation is defined by nine parameters spanning a 9D space $\mathcal{U} = (\mathcal{U}_T, \mathcal{U}_R, \mathcal{U}_S)$. Three parameters are required to define the location $\vec{T} = (t_x, t_y, t_z)$, $\vec{T} \in \mathcal{U}_T$, three to capture the orientation $\vec{R} = (\phi_x, \phi_y, \phi_z)$, $\vec{R} \in \mathcal{U}_R$, and three to define the anisotropic scale $\vec{S} = (s_x, s_y, s_z)$, $\vec{S} \in \mathcal{U}_S$. Even with a very coarse discretization of $d = 10$ possible outcomes for each parameter, the number of hypotheses will be as high as $d^9 = 1{,}000{,}000{,}000$, virtually impossible to evaluate on any current consumer machine. Using statistical shape modeling, also the nonrigid shape of the object can be represented parametrically by considering the coefficients $c_1, c_2, \ldots, c_K$ of the major deformation modes, with K sufficiently large to closely approximate the true shape boundary. In this case the parameter space expands to $\mathcal{U} = (\mathcal{U}_T, \mathcal{U}_R, \mathcal{U}_S, c_1, c_2, \ldots, c_K)$, leading to an explosion in the number of sample hypotheses. In addition, focusing only on the feature extraction in each of these spaces, we face the challenge of ensuring an efficient feature computation also under challenging transformations such as arbitrary orientations or scales, without explicitly transforming the data. All these challenges motivate a drive toward alternative non-learning based boundary delineation methods (see, for example, [7,17,20,21,19,18,22,23]) since they do not require large image databases for the model estimation. However, such methods typically suffer from a series of limitations in terms of robustness to noise and generalization, for which machine learning lends itself as an elegant solution.

3.3 METHODOLOGY

In this section we present a solution to these challenges, a novel feature-learning-based framework for parsing volumetric images split in a two-stage approach: anatomical object localization and nonrigid shape estimation.

3.3.1 PROBLEM FORMULATION

In our model we reformulate the detection and segmentation of an object to a patch-wise classification task described by a set of m parametrized input patches $\vec{X}$ (i.e. observations as regions of interest, image intensities extracted within translated, rotated and scaled boxes) with a corresponding set of class assignments $\vec{y}$, with $y_i = 1$ if the sought anatomical structure is contained in the patch and $y_i = 0$ otherwise,

$1 \leq i \leq m$. In this work, the classifiers are fully connected neural networks, meaning that the size of the filters is equal to the size of the underlying representations. We can define a deep fully connected DNN with the parameters $(\vec{w}, \vec{b})$, where $\vec{w} = (\vec{w}_1, \vec{w}_2, \ldots, \vec{w}_n)^\top$ represents the parameters of all n concatenated kernels over the layers of the network, i.e. the weighted connections between neurons, and $\vec{b}$ encodes the biases of the neurons. The response of a neuron indexed k in the network is defined as:

$$o_k = \delta\left(x_k^\top w_k + b_k\right), \tag{3.1}$$

where δ represents a nonlinear activation function, w_k the weights of incoming connections, x_k the activations of the connected neurons from the previous layer and b_k the bias of the neuron.

The activation function δ is used as a nonlinear operator to synthesize the input information. Possible alternatives are the rectified linear units (ReLU) [24], the hyperbolic tangent or the sigmoid function. For our experiments we use the sigmoid function defined as $\delta(y) = 1/(1 + e^{-y})$, building through our network a multivariate logistic regression model for classification. We define the network response function $\mathcal{R}(\cdot\,; \vec{w}, \vec{b})$ as a probabilistic approximation of the supervised class label for any given example, $\hat{y}^{(i)} = \mathcal{R}(x^{(i)}; \vec{w}, \vec{b})$, where $1 \leq i \leq m$. This is equivalent to estimating the conditional probability density function $p(y^{(i)}|x^{(i)}; \vec{w}, \vec{b})$. Since the learning setup is fully supervised and the input observations are independent, we can use the Maximum Likelihood Estimation (MLE) method:

$$\left(\hat{\vec{w}}, \hat{\vec{b}}\right) = \arg\max_{\vec{w},\vec{b}} \mathcal{L}(\vec{w}, \vec{b}; \vec{X}) = \arg\max_{\vec{w},\vec{b}} \prod_{i=1}^{m} p(y^{(i)}|\vec{x}^{\,(i)}; \vec{w}, \vec{b}), \tag{3.2}$$

where m represents the number of training samples. This is equivalent to minimizing a cost function $\mathcal{C}(\cdot)$ measuring the discrepancy between the network prediction and the expected output, i.e. the true label. We choose the L_2-norm as opposed to the cross-entropy classification loss given that the smoothness of this norm around the ground-truth ensures a more robust aggregation of the most probable hypotheses to a final detection result. As such, we obtain the following minimization problem:

$$\left(\hat{\vec{w}}, \hat{\vec{b}}\right) = \arg\min_{\vec{w},\vec{b}} \left[\mathcal{C}(\vec{X}; \vec{w}, \vec{b}) = \|\mathcal{R}(\vec{X}; \vec{w}, \vec{b}) - \vec{y}\,\|_2^2\right]. \tag{3.3}$$

This can be solved with a standard Stochastic Gradient Descent (SGD) approach [13].

3.3.2 SPARSE ADAPTIVE DEEP NEURAL NETWORKS

The straightforward application of typical neural networks is however not feasible in the volumetric setting. To enable this we propose sparse adaptive deep neural networks, selecting *sparsity* as a means to simplify the neural network, aiming for computational efficiency and to avoid overfitting (see Fig. 3.2). We achieve this by

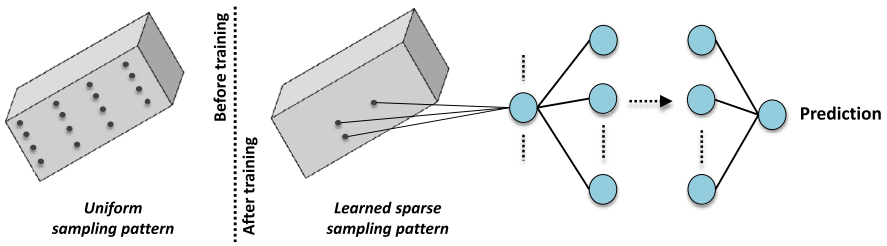


FIGURE 3.2

Visualization of the difference between uniform/handcrafted feature patterns and self-learned, sparse, adaptive patterns.

Algorithm 3.1 Learning algorithm with iterative threshold-enforced sparsity

Pre-training: $\vec{w}^{(0)} \leftarrow \vec{w}$ (small # epochs)
Set sparsity map $\vec{s}^{\,(0)} \leftarrow \vec{1}$
$t \leftarrow 1$
for training round $t \leq T$ **do**
    for all filters i with sparsity **do**
        $\vec{s}_i^{\,(t)} \leftarrow \vec{s}_i^{\,(t-1)}$
        Update $\vec{s}_i^{\,(t)}$ – remove smallest active weights
        $\vec{w}_i^{(t)} = \vec{w}_i^{(t-1)} \odot \vec{s}_i^{\,(t)}$
        Normalize active coefficients s.t. $\|\vec{w}_i^{(t)}\|_1 = \|\vec{w}_i^{(t-1)}\|_1$
    end for
    $\vec{b}^{(t)} \leftarrow \vec{b}^{(t-1)}$
    Train network on active weights (small # epochs)
    $t \leftarrow t + 1$
end for
Sparse kernels: $\vec{w}_s \leftarrow \vec{w}^{(T)}$
Bias values: $\vec{b}_s \leftarrow \vec{b}^{(T)}$

permanently eliminating filter weights, while approximating as well as possible the original filter response. In general terms, we aim to find a sparsity map $\vec{s}$ for the network weights $\vec{w}$, such that over T training rounds, the response residual ϵ given by:

$$\epsilon = \|\mathcal{R}(X; \vec{w}_s, \vec{b}_s) - \vec{y}\,\|_2^2 \tag{3.4}$$

is minimal, where $\vec{b}_s$ denotes the biases of neurons in the sparse network and $\vec{w}_s$ denotes the learned sparse weights, determined by the sparsity map $\vec{s}$ with $s_i \in \{0, 1\}, \forall i$. The learning is based on a greedy iterative process in which we gradually eliminate neural connections with minimal impact on the network response, while continuing the training on the remaining active connections (see Algorithm 3.1).

In each round $t \leq T$ a subset of active connections with minimal absolute value is selected and permanently removed from the network. In order to restore the network response to a certain degree, we re-normalize each filter to preserve its L_1-norm (see Algorithm 3.1). The training is then continued on the remaining active connections, driving the recovery of the neurons from the missing information (see step 12 of Algorithm 3.1):

$$\left(\hat{\vec{w}}^{(t)}, \hat{\vec{b}}^{(t)}\right) = \arg \min_{\substack{\vec{w}: \vec{w}^{(t)} \\ \vec{b}: \vec{b}^{(t)}}} \mathcal{C}(\vec{X}; \vec{w}, \vec{b}), \tag{3.5}$$

where $\vec{w}^{(t)}$ and $\vec{b}^{(t)}$ (computed from the values in round $t-1$) are used as initial values in the optimization step.

Our system learns on the lowest level adaptive, sparse data patterns which capture essential structures in the data, explicitly discarding input with minimal impact on the network response function $\mathcal{R}$ (see Fig. 3.2). The computed patterns can reach sparsity levels of **90–95%**, meaning that only 5–10% of the weights of the low-level, fully-connected kernels can approximate the original network response $\mathcal{R}$. This not only increases the evaluation speed by around 2 orders of magnitude, but also acts as regularization for the model and reduces the likelihood of overfitting during training. We call this type of networks Sparse Adaptive Deep Neural Networks (SADNN).

3.3.3 MARGINAL SPACE DEEP LEARNING

In the context of volumetric image parsing we propose to use SADNN to estimate the transformation parameters. Assuming a restricted affine transformation, let $\vec{T} = (t_x, t_y, t_z)$, $\vec{R} = (\phi_x, \phi_y, \phi_z)$ and $\vec{S} = (s_x, s_y, s_z)$ define the translation, orientation, and anisotropic scaling of an anatomical object. Given an observed input image I, the estimation of the transformation parameters is equivalent to maximizing the posterior probability:

$$\left(\hat{\vec{T}}, \hat{\vec{R}}, \hat{\vec{S}}\right) = \arg \max_{\vec{T}, \vec{R}, \vec{S}} p(\vec{T}, \vec{R}, \vec{S}|I). \tag{3.6}$$

Using a classifier to directly approximate this distribution is not feasible given the large dimensionality of the parameter space. The mechanism of Marginal Space Learning (MSL) [5] proposes to split this space in a hierarchy of clustered, high-probability regions of increasing dimensionality starting in the position space, extending to the position–orientation space, and finally to the full 9D space, including also the anisotropic scaling information of the object. This can be easily achieved based on the following factorization:

$$\begin{aligned} \left(\hat{\vec{T}}, \hat{\vec{R}}, \hat{\vec{S}}\right) &= \arg \max_{\vec{T}, \vec{R}, \vec{S}} p(\vec{T}|I) p(\vec{R}|\vec{T}, I) p(\vec{S}|\vec{T}, \vec{R}, I) \\ &= \arg \max_{\vec{T}, \vec{R}, \vec{S}} p(\vec{T}|I) \frac{p(\vec{T}, \vec{R}|I)}{p(\vec{T}|I)} \frac{p(\vec{T}, \vec{R}, \vec{S}|I)}{p(\vec{T}, \vec{R}|I)}, \end{aligned} \tag{3.7}$$

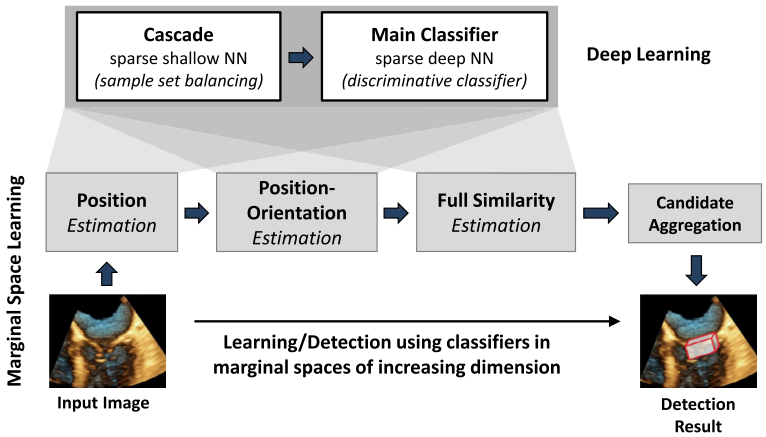


FIGURE 3.3

Schematic visualization of the marginal space deep learning framework.

where the probabilities $p(\vec{T}|I)$, $p(\vec{T}, \vec{R}|I)$, and $p(\vec{T}, \vec{R}, \vec{S}|I)$ are defined in the previously enumerated spaces, also called *marginal spaces* (see Fig. 3.3). In our method we directly approximate each of these distributions with an SADNN, $\mathcal{R}(X; \vec{w}_s, \vec{b}_s)$. This reduces the problem to learning classifiers in these marginal spaces and then scanning each space exhaustively to estimate the object transformation. The workflow starts with learning the translation parameters in the translation space $\mathcal{U}_T(I)$. After this only the positive hypotheses with highest probability, clustered in a dense region are augmented with discretized orientation information, to build the joint translation–orientation space $\mathcal{U}_{TR}(I)$. The same principle applies when extending to the full 9D space $\mathcal{U}_{TRS}(I)$. The end-to-end optimization is defined by:

$$\begin{aligned}
\left(\hat{\vec{T}}, \mathcal{U}_{TR}(I)\right) &\leftarrow \arg\max_{\vec{T}} \mathcal{R}\left(\mathcal{U}_T(I); \vec{w}_s, \vec{b}_s\right) \\
\left(\hat{\vec{T}}, \hat{\vec{R}}, \mathcal{U}_{TRS}(I)\right) &\leftarrow \arg\max_{\vec{T}, \vec{R}} \mathcal{R}\left(\mathcal{U}_{TR}(I); \vec{w}_s, \vec{b}_s\right) \\
\left(\hat{\vec{T}}, \hat{\vec{R}}, \hat{\vec{S}}\right) &\leftarrow \arg\max_{\vec{T}, \vec{R}, \vec{S}} \mathcal{R}\left(\mathcal{U}_{TRS}(I); \vec{w}_s, \vec{b}_s\right),
\end{aligned} \tag{3.8}$$

where $\mathcal{R}(\cdot\,; \vec{w}_s, \vec{b}_s)$ denotes the response of each of the sparse adaptive deep neural networks, learned from the supervised training data $(\vec{X}, \vec{y})$ in each marginal space. Using this type of hierarchical parametrization brings a speed-up of 6 orders of magnitude compared to the exhaustive search in the 9D space (see proof in [5]). The pipeline is visualized in Fig. 3.3.

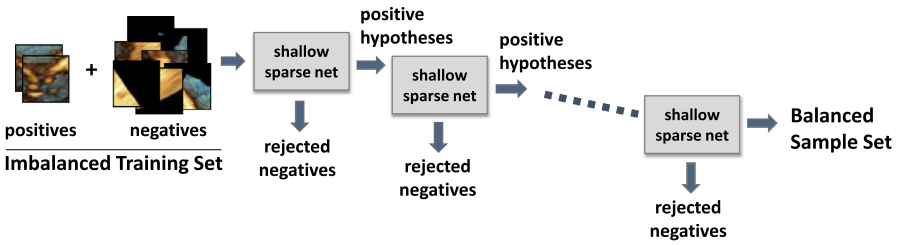


FIGURE 3.4

Schematic visualization of the negative-sample filtering cascade.

Algorithm 3.2 Negative sample filtering algorithm

P – set of positive samples
N – set of negative samples ($|P| << |N|$)
while $|N| \geq 1.5 \times |P|$ **do**
    Learn shallow SADNN using Algorithm 3.1
    $d \leftarrow$ largest decision boundary with FNR $= 0$
    $N' \leftarrow$ filter N based on d – eliminate true negatives
    $N \leftarrow N'$
end while

3.3.3.1 Cascaded Negative Filtering

One important particularity of the learning task in each marginal space is the high class-imbalance which can reach ratios of $1 : 1000$ positive to negative samples. This impacts not only the training efficiency but also the stochastic sampling of the gradient. The main observation is that typically a large percentage of the negative hypotheses are very easy to classify. As such, we propose to use a cascade of shallow neural networks to efficiently and effectively filter the negative hypotheses prior to using a deep powerful model for classification. In each stage of the cascade we train a shallow, sparse neural network and adaptively tune its decision boundary to eliminate as many true negative hypotheses from the training set as possible. The remaining hypotheses, classified as positives, are propagated to the next stage of the cascade where the same filtering step is applied until the training set is balanced (see Fig. 3.4 and Algorithm 3.2).

Using this mechanism the size of the sample set processed by the main classifier in each stage is reduced from $|N|+|P|$ to approximately $3\times|P|$, improving the scanning performance by an additional 2 orders of magnitude (depending on the imbalance ratio).

3.3.3.2 Nonrigid Deformation Estimation

Based on the underlying image information and the initial mean shape computed based on the ground-truth data, we propose an active shape model to guide the shape

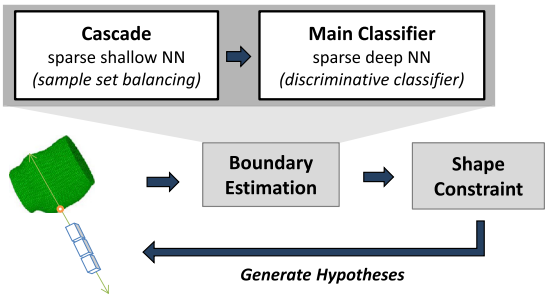


FIGURE 3.5

Schematic visualization of the boundary deformation with SADNN. Starting from the current shape, the SADNN is aligned and applied along the normal for each point of the mesh, the boundary is deformed and projected under the current shape space. The process is iteratively repeated.

deformation. Since the original approach based on energy deformation [7] is not feasible in a 3D setup where the image information and the boundary context are very complex, machine learning has been used as an alternative mechanism to guide the object boundary in both 2D [25,26] and 3D [5]. Typically, a classifier labeled as boundary detector is trained at specific anatomical locations to detect the shape boundary based on local evidence. Following this approach we propose to use the SADNN (with negative filtering cascade) as a boundary classifier to automatically learn adaptive, sparse feature sampling patterns directly from low-level image data. Essentially, for a given control point of the warping mesh, we are dealing with the same problem as faced in the joint translation–orientation space, in the localization stage. Training is performed by using positive samples on the current ground-truth boundary for each mesh point (aligned with the corresponding normal) and using negative samples at various distances from the boundary. Our experiments show that the sparse adaptive patterns are essential in efficiently and effectively capturing the relevant anatomical structures around the boundary. The iterative process is illustrated in Fig. 3.5. The boundary estimation is followed by constraining the deformed shape to the space of shapes corresponding to the current object. We use statistical shape modeling for the constraint, where we estimate from the training set the linear shapes subspace through principal components analysis and online we project the current shape into this subspace using the learned linear projector. The process of boundary estimation and shape constraint enforcement are iteratively applied for a number of pre-determined iterations or until there are no large deformations.

3.3.4 AN ARTIFICIAL AGENT FOR IMAGE PARSING

While the proposed framework systematically addresses many limitations of the state-of-the-art, the algorithmic steps remain unconstrained, exhaustive, and thus sub-

optimal in terms of runtime. For example, the scanning is always performed on the entire parameter space at testing time, even if the positive hypotheses are typically clustered in dense regions. To address this limitation, a principled more intelligent mechanism is required. In general, there is no precise definition or quantification of intelligence or intelligent behavior [27–29]. We can argue that intelligence represents the ability of an algorithm or artificial entity to learn and understand tasks, as opposed to mechanically following predefined solution steps [30]. Relating this to the introduced context of image parsing using machine learning the most important limitation of our approach and other state-of-the-art solutions is the fact that the modeling of object appearance and the search for the optimal transformation parameters are completely decoupled steps.

In the following, we make a first step toward self-taught virtual agents for image understanding in the context of medical image parsing, by formulating the landmark detection problem as a generic learning task for an artificial agent. Our solution leverages state-of-the-art representation learning techniques through deep learning [1] and powerful solutions for generic behavior learning through reinforcement learning [12] to create a model encapsulating a cognitive-like learning process to discover strategies for localizing arbitrary landmarks, using only the raw input image information and the landmark annotations. In other words, we do not use the machine to execute predefined solution steps, but give the machine the ability to find itself the solution.

3.3.4.1 Cognitive Modeling Using Deep Reinforcement Learning

Building powerful artificial agents that can emulate or even surpass human performance at given tasks requires the use of an automatic, generic learning model observed not only in exploratory, unsupervised human cognition [30], but also in basic reward-based animal learning methods [31]. The artificial agent needs to be equipped with at least two fundamental capabilities found at the core of the human and animal intelligence. At perceptual level is the automatic capturing and disentangling of high-dimensional signal data which describes the complete situation in which the agent can find itself, while on cognitive level is the ability to reach decisions and act upon this entire observed information flow [30]. While deep learning can be used to process and interpret the perceived stream of data, reinforcement learning lends itself as a powerful tool for cognitive modeling.

Reinforcement learning (RL) is a technique aimed at effectively describing learning as an end-to-end cognitive process, instead of as a predefined methodology [32]. A typical RL setting is composed by an artificial agent that can interact with an uncertain environment with the target of reaching pre-determined goals. The agent can observe the state of the environment and choose to act on it, similar to a trial-and-error search [32], maximizing the future reward signal received as a response from the environment (see Fig. 3.6 for a schematic overview). This reward-based decision process is modeled in RL theory as a *Markov Decision Process* (MDP) [32] defined by a tuple $\mathcal{M} := (\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma)$, where:

- $\mathcal{S}$ represents a finite set of states, $s_t \in \mathcal{S}$ being the state of the agent at time t

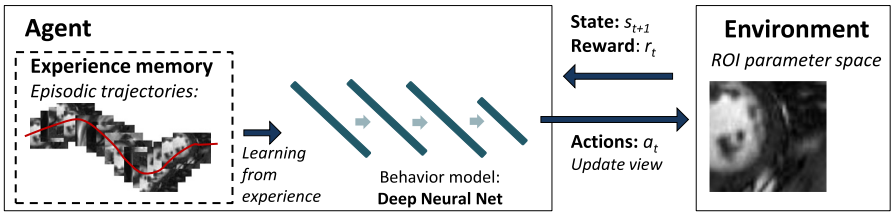


FIGURE 3.6

System diagram showing the interaction between the artificial agent and the environment during training, in the context of landmark detection. The state s_t of the environment at time t is defined by the current view of the agent, given as a fixed patch. Depending on the current state, the actions of the agent directly impact the environment, resulting in a new state and a quantitative reward, (s_{t+1}, r_t). The experience memory stores the last states visited by the agent, which are randomly sampled to periodically update the parameters of the neural network.

- $\mathcal{A}$ represents a finite set of actions allowing the agent to interact with the environment, $a_t \in \mathcal{A}$ being the action the agent performs at time t
- $\mathcal{T} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0; 1]$ is a stochastic transition function, where $\mathcal{T}_{s,a}^{s'}$ describes the probability of arriving in state s' after the agent performed action a in state s
- $\mathcal{R} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow \mathbb{R}$ is a scalar reward function, where $\mathcal{R}_{s,a}^{s'}$ denotes the expected reward after a state transition
- γ is the discount factor controlling the importance of future versus immediate rewards.

Formally, the future discounted reward of an agent at time $\hat{t}$ can be written as $R_{\hat{t}} = \sum_{t=\hat{t}}^{T} \gamma^{t-\hat{t}} r_t$, with T marking the end of a learning episode and r_t defining the immediate reward the agent receives at time t. Especially in model-free reinforcement learning, the target is to find the optimal so-called action-value function $Q^*(s, a)$, denoting the maximum expected future discounted reward when starting in state s and performing action a:

$$Q^*(s, a) = \max_{\pi} \mathbb{E}\left[R_t | s_t = s, a_t = a, \pi\right], \tag{3.9}$$

where π is an action policy, or in other words, a probability distribution over actions in each given state. Once the optimal action-value function is estimated, the optimal action policy, determining the behavior of the agent, can be directly computed in each state as:

$$\forall s \in \mathcal{S} : \pi^*(s) = \arg\max_{a \in \mathcal{A}} Q^*(s, a). \tag{3.10}$$

One important relation satisfied by the optimal action-value function Q^* is the Bellman optimality equation [33] which represents a recursive formulation of Eq. (3.9).

This is defined as:

$$\begin{aligned} Q^*(s,a) &= \sum_{s'} \mathcal{T}_{s,a}^{s'} \left(\mathcal{R}_{s,a}^{s'} + \gamma \max_{a'} Q^*(s',a') \right) \\ &= \mathbb{E}_{s'} \left(r + \gamma \max_{a'} Q^*(s',a') \right), \end{aligned} \tag{3.11}$$

where s' defines a possible state visited after s, a' the corresponding action and $r = R_{s,a}^{s'}$ represents a compact notation for the current, immediate reward. Viewed as an operator τ, the Bellman equation defines a contraction mapping. Theoretical results [33] show that by iteratively applying $Q_{i+1} = \tau(Q_i), \forall(s,a)$, the function Q_i converges to Q^* at infinity. This standard policy iteration approach is however not feasible in practice. An alternative is to use model-free temporal difference methods, typically Q-Learning [15], which exploit correlations of consecutive states. A step further toward a higher computational efficiency is the use of parametric functions to approximate the Q-function [34]. Considering the expected nonlinear structure of the action-value function [15], neural networks represent a potentially powerful approximation solution [35,16,36,37]. The first end-to-end reinforcement learning approach making use of state-of-the-art deep neural networks for direct policy approximation is presented by Mnih et al. [12].

In the following we will show how these reinforcement learning concepts can be applied in the context of intelligent image parsing, grounded on recent advances in deep learning. Fig. 3.6 gives an overview of our approach.

3.3.4.2 Learning to Scan for Anatomical Objects

Let us focus on the problem of landmark detection. The principles can easily be extended to different parametric spaces as modeled in marginal space learning. Given a set of N training images $I_1, I_2, \ldots, I_N$, each containing M annotated landmarks. Focusing on one particular landmark indexed in each training example, we would like to train an intelligent agent that can automatically discover strategies for finding the selected landmark as opposed to following tedious scanning routines.

3.3.4.2.1 Problem Formulation

We reformulate our task as a *Markov Decision Process* $\mathcal{M} := (\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma)$. In the following we will specify the state and action spaces and define the reward system (note that the transition probabilities $\mathcal{T}$ are unknown in our model-free approach).

States. The agent needs to be able to capture its surrounding environment and be able to react upon what is observed. In consequence, states need to be discriminative and self-describing. Following this requirement, the selection of the state space is inspired by principles found in the animal and human visual perception system [38]. When analyzing an image or a static scene, with the focus set on one particular point, the surrounding context which is actively perceived is very limited. In other words, what is captured is dense local information around the focus point and limited

global context from the surrounding neighborhood [38]. Starting from this observation, we make a similar locality assumption and define a state observed at time t as $s_t = (I_t, x_t, y_t, l_t)$, i.e. a local patch of size $l_t \times l_t$ centered at position (x_t, y_t) in the observed image I_t. States which are close to the target landmark location will directly capture the position of the landmark in their context. For distant states, the relation to the landmark location is intrinsic, captured indirectly by information from the context of the current patch. This is consistent with the general properties of the perception model introduced in [38].

Actions. In each state the agent interacts with the enclosing environment by performing certain actions from a predefined action set [15]. The set of actions is chosen in such a way that the agent is given the possibility to explore the entire environment. Located in state s_t at time t, with the aim of reaching the location of a target landmark, we give the agent the possibility to perform one of the following discrete actions: move *upward*, *downward*, *left* or *right*. For each action, we simplify to a single-pixel move: $x_{t+1} \leftarrow x_t \pm 1$ and $y_{t+1} \leftarrow y_t \pm 1$. Once the target has been reached, no further action is performed and the search is finished. At testing time this results in an oscillatory transition between neighboring states at the target, eliminating the need for an additional *stop* action.

Rewards. The reward system is based on the change in relative position at state $s_t : (x_t, y_t)$ with respect to the target position of the landmark $s_{target} : (x_{target}, y_{target})$. Intuitively, for a move in the correct direction, a positive reward proportional to the target-distance reduction is given, whereas a move in the wrong direction is punished by a negative reward of equal magnitude. Formally, the reward at time t is given by:

$$r_t = dist(s_t, s_{target}) - dist(s_{t+1}, s_{target}). \tag{3.12}$$

The only exception to this rule is an attempt to leave the image field by crossing the image border. Such an action is always given the highest punishment of -1. As it can be observed, the reward is always correlated with the goodness of a performed action. The motivation for this more complex reward system compared to the simple ± 1 reward used by Mnih et al. [12] is inspired by principles applied in complex trial–error systems [30], simulating more closely human experience. In other words, good actions, contributing significantly toward reaching the goal, are given a high reward, whereas actions that only marginally improve the state of the agent receive little reward.

3.3.4.2.2 Proposed Method

Given the environment definition and reward system, the goal of the agent is to select actions by repeatedly interacting with the enclosing environment in order to maximize cumulative future reward (see Eq. (3.9)). This optimal behavior is defined by the optimal policy π^* selected from the space of all possible policies $\pi \leftarrow p(action|state)$. As shown in Eq. (3.10), finding the optimal policy is straightforward given the optimal action-value function Q^*.

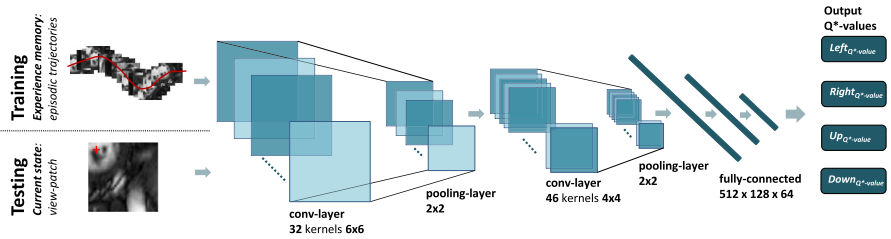


FIGURE 3.7

Schematic illustration of the 7-layer deep convolutional neural network used to approximate the optimal action-value function $Q^*(s_t, a_t)$. The network output is the Q^* value, the maximum expected future reward for all possible actions in the current state (moving *left*, *right*, *up*, or *down*). During the training stage we apply experience replay by randomly sampling patches from all episodic trajectories stored in memory to define the mini-batch used to update the network parameters. During the testing stage the input to the network is one single patch, the current state, or in other words, the current view of the agent on the image. Evaluating the network on this patch helps the agent decide in which direction to navigate in order to reach the target location (visualized as a red +).

In this work we propose a model-free, temporal difference approach introduced in the context of game learning by Mnih et al. [12], using a deep convolutional neural network (CNN) to approximate the optimal action-value function Q^*. Defining the parameters of a deep CNN as $\theta = [\,\vec{w}, \vec{b}\,]$, we use this architecture as a generic, non-linear function approximator $Q(s, a; \theta) \approx Q^*(s, a)$, called deep Q network (DQN), which we visualize in Fig. 3.7. To account for the possible divergence issues during training we apply the concepts of *reference update-delay* and *experience replay*.

Similar to the temporal difference Q-Learning algorithm [15], a deep Q network can be trained in a reinforcement learning setup using an iterative approach to minimize the mean squared error based on the Bellman optimality criterion (see Eq. (3.11)). At any iteration i, we can approximate the optimal expected target values using a set of reference parameters $\theta_i^{ref} := \theta_j$ from a previous iteration $j < i$ as:

$$y = r + \gamma \max_{a'} Q(s', a'; \theta_i^{ref}). \tag{3.13}$$

As such we obtain a sequence of well-defined optimization problems driving the evolution of the network parameters. The error function at each step i is defined as:

$$\theta_i = \min_{\theta_i} \mathbb{E}_{s,a,r,s'} \left[(y - Q(s, a; \theta_i))^2 \right] + \mathbb{E}_{s,a,r} \left[\mathbb{V}_{s'}[y] \right]. \tag{3.14}$$

This is a standard, supervised setup for deep learning for which mini-batch gradient-based approaches [13] combined with backpropagation [39] represent a powerful solution. In our framework we periodically apply stochastic gradient descent steps,

approximating the gradient by randomly sampling the gradient function, given as:

$$\nabla_{\theta_i} = \mathbb{E}_{s,a,r,s'}\left[(y - Q(s,a;\theta_i))\,\nabla_{\theta_i} Q(s,a;\theta_i)\right]. \tag{3.15}$$

Reference Update-Delay. As motivated in [12], using a different network to compute the reference values for training brings robustness to the algorithm. In such a setup, changes to the current parameters θ_i and implicitly to the current approximator $Q(\cdot\,;\theta_i)$ cannot directly impact the reference output y, introducing an update-delay and thereby reducing the probability to diverge and oscillate in suboptimal regions of the optimization space [12]. Fig. 3.7 depicts the used architecture providing a different, more in depth visualization of the main system diagram introduced in Fig. 3.6. In a given state, specified by the current view-patch, we evaluate the neural network on that particular patch to simultaneously obtain Q^*-value estimates for all possible actions: $Q^*(s,a_1)$, $Q^*(s,a_2)$, ... Given the estimates, the agent applies an ϵ-greedy policy, choosing a random action with probability ϵ and following the current policy estimation (choosing the action with the maximum future, discounted reward) with probability $1-\epsilon$. During learning a value decay is applied on the parameter ϵ reaching a trade-off between an effective space exploration and a greedy, consistent policy exploitation strategy.

Experience Replay. To ensure the robustness of the parameter updates and train more efficiently, we propose to use the concept of experience replay [40]. In experience replay, the agent stores a limited amount of previously visited states (typically the last states) in a so-called experience memory and then samples that memory to update the parameters of the underlying neural network [16,12]. In our case, we learn in a sequence of episodes which quantify the local performance of the agent on given training images. Before the start of one episode, the agent is given a random image from the complete training set and a random start-state, i.e. start position in that image. During the course of the episode, the agent performs actions applying the ϵ-greedy behavior policy and navigating through this local environment (the given training image). The episode finishes when the target state is reached, or in other words, the landmark location is found, or a pre-defined maximum number of actions are executed. This defines a so-called *trajectory* t_i (we also call it episodic trajectory) in the image space which encodes the applied search strategy as a sequence of visited states. All these trajectories are stored in our replay memory since they represent the entire experience the agent has accumulated on different images. In our experiments we store the last P trajectories as $\mathcal{E} = [t_1, t_2, \ldots, t_P]$. At fixed intervals during training (typically every 4–6 state transitions) a parameter update is performed using a random mini-batch of states extracted from $\mathcal{E}$. Similarly to [40,16,12], we argue that this kind of approach is essential for ensuring the training convergence. Updating the deep neural network on locally correlated states (similar to Q-Learning) does not generalize; on the contrary, this negatively affects the performance of the network in other parts of the state-space. Using a uniformly sampled set of previous experiences, averages the distribution of the network input, reducing oscillations, and ensuring a

much faster and robust learning experience. Figs. 3.6 and 3.7 visualize this type of experience-based learning approach.

3.4 EXPERIMENTS

In this section we present the experiments we performed for validation. These include object detection and segmentation in 3D and landmark detection in 2D and 3D.

3.4.1 ANATOMY DETECTION AND SEGMENTATION IN 3D

In order to evaluate the performance of the MSDL framework, we compare against the state-of-the-art MSL solution [5] on detecting and segmenting the aortic valve (root) in 3D transesophageal echocardiogram (TEE) images. The aortic root connects the ascending aorta to the left ventricular outflow tract and is represented through a tubular grid (see Fig. 3.9). This is a particularly challenging task, considering the high variation in anatomical appearance of the valve, the heart-motion, and implicitly motion of the valve-leaflets, as well as the image quality limitations, i.e. noise in ultrasound images.

3.4.1.1 Dataset

The dataset used for evaluation stems from 869 patients and contains 2891 3D TEE volumes. The size of the images vary from $100 \times 100 \times 50$ to $250 \times 250 \times 150$ voxels while the spatial resolution lies between 0.75 to 1 mm. The preprocessing step includes the intensity normalization of the image as well as the resampling of the volumes to an isotropic 3 mm resolution.

The validation setup is the same for both approaches and is based on a random split of the volumes at patient level in 2481 training and 410 testing volumes (about 84% and 16%). At patient level this results in 719 patients in the training set and 150 patients in the test set. The ground-truth data was obtained through manual annotation performed by experts [41]. Note that the pose of the 3D bounding box is fully determined by the underlying anatomy.

3.4.1.2 Model Selection and Training Parameters

All meta-parameters related to the sub-space sampling and learning model are optimized using a systematic grid-search. Across all 3 marginal spaces we use the same SADNN architecture for the cascade and main classifier, i.e. cascade with 2 layers = 5832 (sparse) $\times$ 60 $\times$ 1, and main classifier with 4 layers = 5832 (sparse) $\times$ 150 $\times$ 80 $\times$ 50 $\times$ 1 hidden units. In all networks we used sigmoid activations.

3.4.1.3 Evaluation

The measures used to quantify the performance of the object localization are the difference in center position and the corner distance error, and the average distance between the 8 corners of the detected box and the ground-truth box. The second value

Table 3.1 Comparison of the performance of the state-of-the-art MSL [5] and the proposed MSDL framework for aortic valve detection. The measures used to quantify the quality of the results w.r.t. to the ground-truth data are the error of the position of the box and mean corner distance (both measured in millimeters). The superior results are displayed in bold

	Position error [mm]				Corner error [mm]			
	Training data		Test data		Training data		Test data	
	MSL	MSDL	MSL	MSDL	MSL	MSDL	MSL	MSDL
Mean	3.12	**1.47**	3.34	**1.83**	5.42	**2.80**	6.16	**3.72**
Median	2.80	**1.27**	3.05	**1.58**	4.98	**2.58**	5.85	**3.34**
STD	1.91	**0.99**	1.85	**1.31**	2.47	**1.23**	2.31	**1.74**

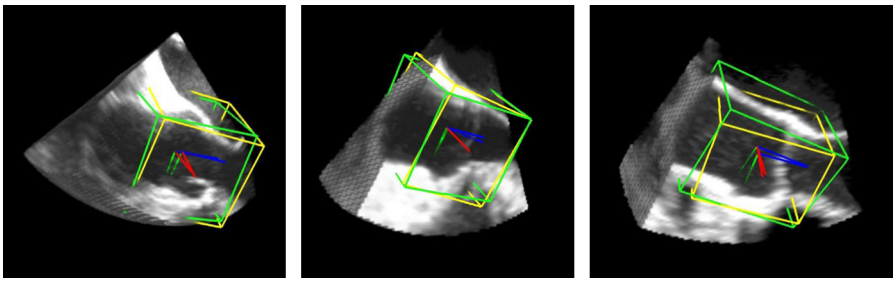


FIGURE 3.8

Example of detected bounding boxes for images from different patients from the test set. The detected box is shown in green and the ground-truth in yellow.

gives also an indication about the accuracy of the orientation and scale estimation. Table 3.1 shows the obtained results. The MSDL framework significantly improves the performance of the state-of-the-art MSL solution, improving the mean position error by 45.2% and the corner distance error by 39.6%. Fig. 3.8 shows qualitative results for different patients from the test set. The MSDL framework matches the excellent run-time performance of the MSL framework running in less than 0.5 seconds using only the CPU. This is greatly due to the use of sparse data sampling patterns which bring a speed-up of 300× to the MSDL-non-sparse, hence also the significant computational benefit of the network simplification. We also emphasize here that using a cascade for the early rejection of hypotheses is essential to reaching this competitive performance. Depending on the imbalance ratio in each marginal space, using a cascade brings around 2 orders of magnitude speed-wise improvement.

For the segmentation experiments the performance is measured by computing the distance between the segmentation and the ground-truth annotation mesh. Table 3.2 shows that the MSDL approach outperforms the MSL method by reducing the average mesh error from 1.04 to 0.9 mm. This represents an improvement of 13.5% at a runtime of less than 1 seconds/volume. Qualitative results can be seen in Fig. 3.9.

Table 3.2 Comparison of the performance of the state-of-the-art MSL [5] and the proposed MSDL framework for aortic valve segmentation. The measure illustrates the distance of the detected mesh to the ground-truth for both the initialization from the detected box as well as the distance after boundary refinement. The superior results are displayed in bold

	Training data, dist. to gt. mesh [mm]				Test data, dist. to gt. mesh [mm]			
	Initialization		Final		Initialization		Final	
	MSL	MSDL	MSL	MSDL	MSL	MSDL	MSL	MSDL
Mean	2.08	**1.16**	1.17	**0.89**	2.06	**1.21**	1.04	**0.90**
Median	1.94	**1.09**	1.05	**0.82**	1.95	**1.10**	0.98	**0.80**
STD	0.83	**0.40**	0.66	**0.35**	0.79	**0.55**	0.50	**0.48**

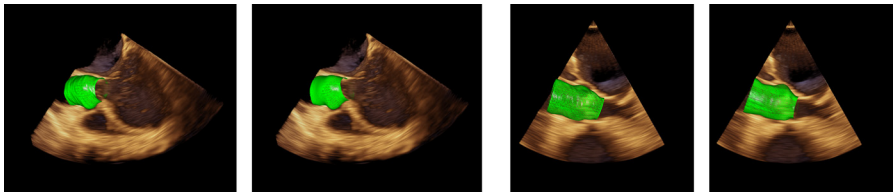


FIGURE 3.9

Example images showing the aortic valve segmentation results for different patients from the test set. The images are organized as two pairs, with the detection on the left and ground-truth on the right.

3.4.1.4 *Additional Experiments and Discussion*

Given the fact that the accuracy of the nonrigid shape segmentation directly depends on the accuracy of the object detection, Table 3.2 is not an optimal indicator for the superiority of the DL-based segmentation alone. In this context we have investigated the performance of two mixed framework-versions: one combining our MSDL object detection with the original segmentation approach from Zheng et al. [5] and the other combining the MSL object detection solution [5] with our the DL-based segmentation. This gives a direct comparison between typical handcrafted features and adaptive sparse patterns in capturing the image context around the boundary. The experiment highlights the overall superiority of our end-to-end DL-based approach on the test set. Table 3.3 shows the results.

We also highlight the performance of our cascaded sparse architecture compared to a single standard end-to-end deep network in terms of speed. We have experimented with using a CNN both in the cascade and as main deep classifier. Our implementation is based on direct calls to the latest cuDNN 4.0 library on top of a high-end GTX980 GPU architecture. We found that reasonable architectures with 1 convolution layer for the cascade (respectively, 3 convolution layers (with pooling and fully-connected) for the main classifier) are about 400–500 times slower than our original solution. For example, in the orientation stage a deep 3D-CNN could pro-

Table 3.3 Comparison of the performance of our DL-based framework with the state-of-the-art solution [5] and the two mixed variants: combining the MSDL box detection with the original segmentation method [5], and the MSL box detection [5] with our DL-based segmentation method. The superior results are displayed in bold

	Segmentation error [mm]		
	Mean	**Median**	**STD**
Ours	**0.90**	**0.80**	**0.48**
Reference [5]	1.04	0.98	0.50
Detection (ours) + Seg. [5]	0.95	0.88	0.48
Detection [5] + Seg. (ours)	1.00	0.90	0.51

cess around 600 hypotheses/second compared to 420,000 processed with an SADNN. When comparing shallow versions of these type of networks for cascade filtering, the SADNN is around 200× faster, processing over 1,200,000 hypotheses/second using only the CPU. In this context, removing the cascade and using a single end-to-end deep CNN can only further increase this performance deficit, both during training and testing. As such, training our framework with standard deep networks and investigating their accuracy becomes infeasible, requiring in the order of thousands of hours of training time.

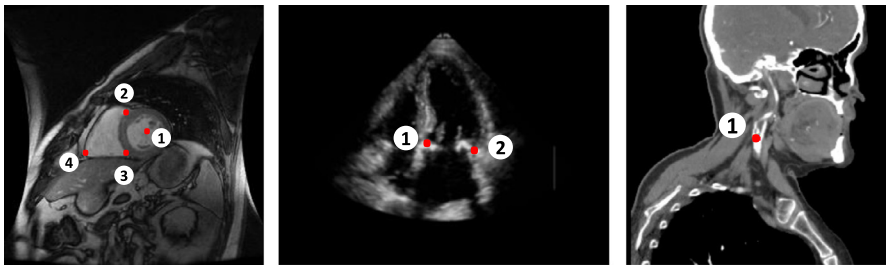
The main reason for this speed difference is not only the cascade filtering but also the data-efficiency of the sparse sampling. Recall that most sparse patterns in our SADNN architecture reach sparsity levels of 90–95%, thus indexing only a small fraction of the data voxels during scanning and minimizing the memory-footprint in the large incoming stream of volumetric data. This is different from the CNN architecture which samples every single voxel multiple times, proportionally to the size of the convolution kernel. However, integrating the CNN in our pipeline in a type of hybrid learning system is part of our ongoing work. For example, we are optimistic in managing to integrate such precise deep architectures into our pipeline to process only small subsets of difficult hypotheses, to further increase the accuracy of the system.

3.4.2 LANDMARK DETECTION IN 2D AND 3D

Accurate landmark detection is a fundamental prerequisite for medical image analysis. In the following we demonstrate the performance of our agent-driven intelligent parsing method on this type of application.

3.4.2.1 Datasets

We perform the evaluation on three datasets. They are composed of 891 short-axis view MR images from 338 patients, 1186 cardiac ultrasound apical four-chamber view images from 361 patients, and 455 head–neck CT scans from 455 patients. The landmarks selected for testing are presented in Fig. 3.10. Each dataset is split



A 2D MRI B 2D US C 3D CT

FIGURE 3.10

Figure A shows the LV-center (1), the anterior/posterior RV-insertion points (2)/(3) and the RV-extreme point (4) in a short-axis cardiac MR image. Figure B highlights the mitral septal annulus (1) and the mitral lateral annulus points (2) in a cardiac ultrasound image and figure C the right carotid artery bifurcation (1) in a head–neck CT scan.

Table 3.4 The detection error on the test sets with superior results highlighted in bold. The error is quantified as the distance to the ground-truth, measured in mm. With * we signify that the results are reported on the same dataset, but on a different training/test data-split than ours

	Detection error [mm]											
	2D-MRI								**2D-US**		**3D-CT**	
	LV-center		**RV-ext**		**RV-post**			**RV-ant**	**M-sep**	**M-lat**	**Bifurc.**	
	Our	**[43]**	**Our**	**[42]**	**Our**	**[43]**	**[42]**	**Our**	**Our**	**Our**	**Our**	**[6]**
Mean	**1.8**	6.2*	**4.9**	8.4*	**2.2**	7.9*	5.9*	**3.7**	**1.3**	**1.6**	**1.8**	2.6
Median	**1.7**	5.4*	**4.2**	5.9*	**1.8**	4.7*	3.9*	**3.0**	**1.2**	**1.3**	**0.8**	1.2
STD	**2.2**	4.0*	**3.6**	16.5*	**1.5**	11.5*	16.0*	**2.3**	**0.8**	**1.4**	**2.9**	5.0

randomly at patient level in approximately 80% training, 10% validation, and 10% testing. The results on the MR dataset are compared to the state-of-the-art results achieved in [42,43] with methods combining context modeling with machine-learning for robust landmark detection. On the CT dataset we compare to [6], a state-of-the-art deep learning solution combined with exhaustive hypotheses scanning. Table 3.4 shows the obtained results.

3.4.2.2 *Evaluation: Learning to Parse*

The training starts from a random initialization of the policy. In a sequence of learning episodes, the agent is given random training images with corresponding random start-states. The agent then follows the ϵ-greedy search strategy in the selected image, generating at the end of the episode a trajectory which is added to its experience memory. During the exploration, periodic updates are applied to the parameters of the

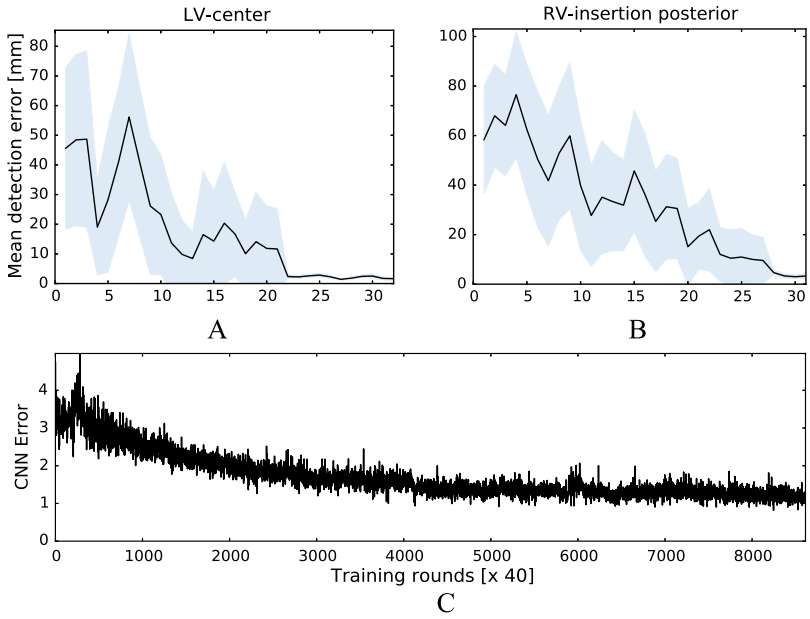


FIGURE 3.11

Plots A and B show the progression of the detection accuracy on the validation set during training. Plot C represents the error of the Bellman optimality equation.

neural network, leading to a more accurate approximation of the optimal Q^* function, given the current experience. This process is repeated in an iterative manner until the detection accuracy on the validation set is minimal. Note that for all experiments the network architecture and training parameters are the same. In terms of training, we propose to use the *rms-prop* [12] mini-batch approach, yielding a better performance than standard stochastic gradient descent. The learning rate is set to $\eta = 0.00025$, justified by the sparse sampling applied in experience replay, while the discount factor is fixed to $\gamma = 0.9$. Other important training parameters are the replay memory size ($P = 100{,}000$ view-patches) and $\epsilon = 0.8$ decaying linearly to 0.05.

Fig. 3.11 shows the performance evolution during training. The high standard deviation of the detection accuracy is correlated with divergent trajectories, given the random initialization of the policy. However, as the policy improves, the detection accuracy increases significantly, reaching the minimum point when all trajectories converge to the correct landmark position. On the second line we show the average expected reward, computed for random states that are kept fixed across the training stages. The plot on the last line in Fig. 3.11 shows the progression of the mean squared error in the Bellman equation. In our experiments we quantify the quality of the learned policy and decide the number of training rounds based on the mean detection error on the cross-validation set.

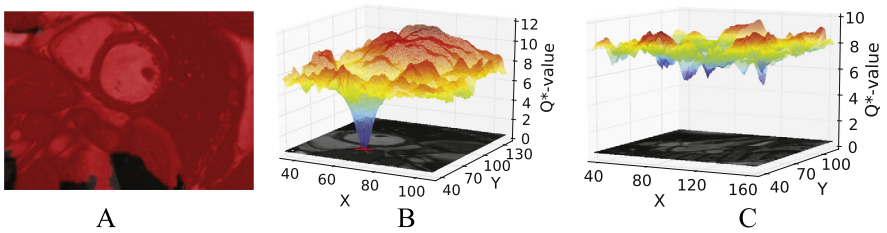


FIGURE 3.12

Figure A highlights in transparent red all the starting positions converging to the correct landmark position. Figures B and C visualize the optimal action-value function Q^*, with each state (image position) encoding the highest expected reward, considering all actions allowed in that state. We also call it the optimal Q^*-field. In the first example, as can be observed, the global minimum point reaches near zero-value at the location of the target landmark (highlighted with a red **x** on the X–Y projected MR image). This is different from the other example where the Q^*-field is visualized for an image that does not contain the target landmark. This clearly indicates the absence of the object.

3.4.2.2.1 Agent Evaluation

During the evaluation the agent starts in a random or predefined state (e.g. expected landmark location based on the ground-truth), and follows the computed policy, iterating through the state space until an oscillation occurs (an infinite loop between two neighboring states). The end state is considered a high-confidence solution for the position of the target landmark, if the expected reward $\max_a Q^*(s_{target}, a) < 1$ (closer than one pixel). If this is not the case, then we consider that the search failed. This gives us a powerful confidence measure for the performance of the agent (see Fig. 3.12B). Using this property we not only detect diverging trajectories, but can also recognize if the landmark is not contained in the image. For this we evaluated trained agents on 100 long-axis cardiac MR images from different patients, showing that in such cases the oscillation occurs at points where the expected future reward is significantly high – at least 4 (see Fig. 3.12C). This suggests the ability of our algorithm to detect when the anatomical landmark is not in the image. Regarding the relation between trajectory convergence and start-state, we observed in randomly selected test images that more than 90% of the possible starting points converge to the correct solution (see Fig. 3.12A). In other words, with only 3 random attempts, the probability of diverging is less than 0.1%. Fig. 3.13 shows examples of trajectories.

3.5 CONCLUSION

In this chapter we presented Marginal Space Deep Learning as an efficient solution for parsing volumetric medical image data. By exploiting the computational efficiency of learning in marginal parameter spaces and the discriminative power

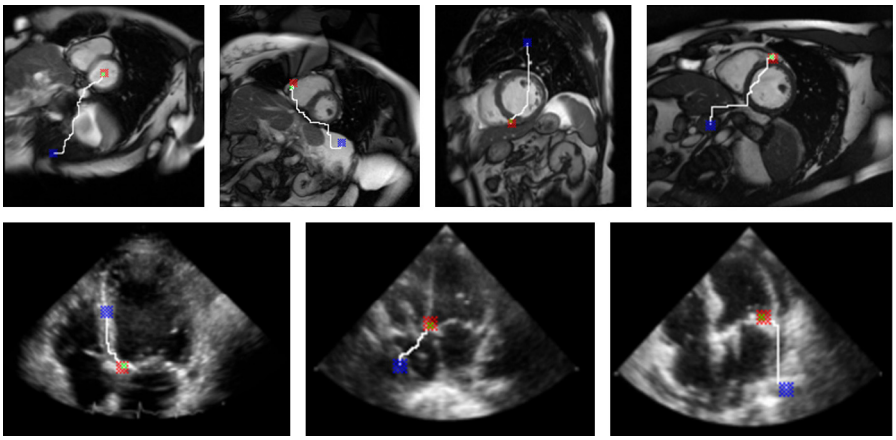


FIGURE 3.13

Example strategies, i.e. trajectories, the considered anatomical landmarks. The blue point denotes the starting position (chosen randomly), the red point the ground-truth position, while green denotes the detection result. The trajectory is visualized as a white curve.

of deep neural network representations, we managed to build a system that significantly outperforms the state-of-the-art. However, the performance is achieved at the cost of tedious scanning efforts that are necessary for the parameter estimation, efforts which limit the scalability of the algorithm to more complex transformations. In this context we proposed an alternative solution that couples the learning of the appearance model and the parameter search as a unified behavioral task for an artificial agent. Instead of following exhaustive search schemes, the algorithm finds an optimal, strategic path in parameter space converging to the optimal location. We showed on the example of landmark detection that this alternative solution achieves accurate results and addresses the computational limitations of exhaustive scanning. Our future work is focused on extending this framework for generic image parsing.

DISCLAIMER

This feature is based on research, and is not commercially available. Due to regulatory reasons, its future availability cannot be guaranteed.

REFERENCES

1. Yoshua Bengio, Aaron C. Courville, Pascal Vincent, Unsupervised feature learning and deep learning: a review and new perspectives, arXiv:1206.5538, 2012.

2. David G. Lowe, Object recognition from local scale-invariant features, in: ICCV, vol. 2, 1999, pp. 1150–1157.
3. Navneet Dalal, Bill Triggs, Histograms of oriented gradients for human detection, in: Conference on Computer Vision and Pattern Recognition, 2005, pp. 886–893.
4. P. Viola, M. Jones, Rapid object detection using a boosted cascade of simple features, in: Conference on Computer Vision and Pattern Recognition, vol. 1, 2001, pp. 511–518.
5. Yefeng Zheng, Adrian Barbu, Bogdan Georgescu, Michael Scheuering, Dorin Comaniciu, Four-chamber heart modeling and automatic segmentation for 3-D cardiac CT volumes using marginal space learning and steerable features, IEEE TMI 27 (11) (2008) 1668–1681.
6. Yefeng Zheng, David Liu, Bogdan Georgescu, Hien Nguyen, Dorin Comaniciu, 3D deep learning for efficient and robust landmark detection in volumetric data, in: Nassir Navab, Joachim Hornegger, M. William Wells, F. Alejandro Frangi (Eds.), MICCAI 2015, Part I, in: Lect. Notes Comput. Sci., vol. 9349, Springer International Publishing, 2015, pp. 565–572.
7. T.F. Cootes, C.J. Taylor, D.H. Cooper, J. Graham, Active shape models – their training and application, Comput. Vis. Image Underst. 61 (1) (1995) 38–59.
8. Florin C. Ghesu, Bogdan Georgescu, Yefeng Zheng, Joachim Hornegger, Dorin Comaniciu, Marginal space deep learning: efficient architecture for detection in volumetric image data, in: Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015 – 18th International Conference, Part I, Munich, Germany, October 5–9, 2015, 2015, pp. 710–718.
9. F.C. Ghesu, E. Krubasik, B. Georgescu, V. Singh, Y. Zheng, J. Hornegger, D. Comaniciu, Marginal space deep learning: efficient architecture for volumetric image parsing, IEEE TMI 35 (5) (2016) 1217–1228.
10. Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, Ruslan Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting, J. Mach. Learn. Res. 15 (2014) 1929–1958.
11. F.C. Ghesu, B. Georgescu, T. Mansi, D. Neuman, J. Hornegger, D. Comaniciu, An artificial agent for anatomical landmark detection in medical images, in: S. Ourselin, L. Joskowicz, M.R. Sabuncu, G. Unal, W. Wells (Eds.), Medical Image Computing and Computer-Assisted Intervention – MICCAI 2016: 19th International Conference, Part III, Athens, Greece, October 17–21, 2016, 2016, pp. 229–237.
12. Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, Demis Hassabis, Human-level control through deep reinforcement learning, Nature 518 (7540) (2015) 529–533.
13. Y. Lecun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proc. IEEE 86 (11) (1998) 2278–2324.
14. Geoffrey E. Hinton, Simon Osindero, Yee-Whye Teh, A fast learning algorithm for deep belief nets, Neural Comput. 18 (7) (2006) 1527–1554.
15. C.J.C.H. Watkins, P. Dayan, Q-learning, Mach. Learn. 8 (3) (1992) 279–292.
16. Martin Riedmiller, Neural fitted Q iteration – first experiences with a data efficient neural reinforcement learning method, in: João Gama, Rui Camacho, Pavel B. Brazdil, Alípio Mário Jorge, Luís Torgo (Eds.), Machine Learning: ECML 2005, in: Lect. Notes Comput. Sci., vol. 3720, Springer, Berlin, Heidelberg, 2005, pp. 317–328.
17. Hans C. van Assen, Mikhail G. Danilouchkine, Alejandro F. Frangi, Sebastián Ordas, Jos J.M. Westenberg, Johan H.C. Reiber, Boudewijn P.F. Lelieveldt, SPASM: a 3D-ASM for

segmentation of sparse and arbitrarily oriented cardiac MRI data, Med. Image Anal. 10 (2) (2006) 286–303.
18. Alison M. Pouch, Hongzhi Wang, Manabu Takabe, Benjamin M. Jackson, Joseph H. Gorman III, Robert C. Gorman, Paul A. Yushkevich, Chandra M. Sehgal, Fully automatic segmentation of the mitral leaflets in 3D transesophageal echocardiographic images using multi-atlas joint label fusion and deformable medial modeling, Med. Image Anal. 18 (1) (2014) 118–129.
19. Robert J. Schneider, Neil A. Tenenholtz, Douglas P. Perrin, Gerald R. Marx, Pedro J. del Nido, Robert D. Howe, Patient-specific mitral leaflet segmentation from 4D ultrasound, in: MICCAI 2011, 2011, pp. 520–527.
20. Steven C. Mitchell, Johan G. Bosch, Boudewijn P.F. Lelieveldt, Rob J. van der Geest, Johan H.C. Reiber, Milan Sonka, 3-D active appearance models: segmentation of cardiac MR and ultrasound images, IEEE Trans. Med. Imaging 21 (9) (2002) 1167–1178.
21. F. Tombari, L. Di Stefano, 3D data segmentation by local classification and Markov random fields, in: Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission, 2011, pp. 212–219.
22. Mustafa Elattar, Esther Wiegerinck, Floortje Kesteren, Lucile Dubois, Nils Planken, Ed Vanbavel, Jan Baan, Henk Marquering, Automatic aortic root landmark detection in CTA images for preprocedural planning of transcatheter aortic valve implantation, Int. J. Cardiovasc. Imaging (2015) 1–11.
23. Alison M. Pouch, Sijie Tian, Manabu Takabe, Jiefu Yuan, Robert Gorman Jr., Albert T. Cheung, Hongzhi Wang, Benjamin M. Jackson, Joseph H. Gorman III, Robert C. Gorman, Paul A. Yushkevich, Medially constrained deformable modeling for segmentation of branching medial structures: application to aortic valve segmentation and morphometry, Med. Image Anal. 26 (1) (2015) 217–231.
24. Xavier Glorot, Antoine Bordes, Yoshua Bengio, Deep sparse rectifier neural networks, in: Geoffrey J. Gordon, David B. Dunson (Eds.), Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, AISTATS-11, J. Mach. Learn. Res. Workshop Conf. Proc. 15 (2011) 315–323, http://www.jmlr.org/proceedings/papers/v15/glorot11a/glorot11a.pdf.
25. B. van Ginneken, A.F. Frangi, J.J. Staal, B.M. ter Haar Romeny, M.A. Viergever, Active shape model segmentation with optimal features, IEEE Trans. Med. Imaging 21 (8) (2002) 924–933.
26. D.R. Martin, C.C. Fowlkes, J. Malik, Learning to detect natural image boundaries using local brightness, color, and texture cues, IEEE Trans. Pattern Anal. Mach. Intell. 26 (5) (2004) 530–549.
27. A.M. Turing, Computing Machinery and Intelligence, 1950.
28. K. Appel, W. Haken, J. Koch, Every planar map is four colorable. Part II: Reducibility, Illinois J. Math. 21 (3) (1977) 491–567.
29. Hee-Jun Park, Byung Kook Kim, Kye Young Lim, Measuring the machine intelligence quotient (MIQ) of human–machine cooperative systems, IEEE Trans. Syst. Man Cybern., Part A, Syst. Hum. 31 (2) (2001) 89–96.
30. Stuart J. Russell, Peter Norvig, Artificial Intelligence: A Modern Approach, second ed., Pearson Education, ISBN 0137903952, 2003.
31. E.L. Thorndike, Animal Intelligence: Experimental Studies, Animal Behavior Series, Macmillan, 1911.
32. Richard S. Sutton, Andrew G. Barto, Introduction to Reinforcement Learning, first ed., MIT Press, 1998.

33. Richard Bellman, Dynamic Programming, first ed., Princeton University Press, Princeton, NJ, USA, 1957.
34. J.N. Tsitsiklis, B. Van Roy, An analysis of temporal-difference learning with function approximation, IEEE Trans. Autom. Control 42 (5) (1997) 674–690, http://dx.doi.org/10.1109/9.580874.
35. Brian Sallans, Geoffrey E. Hinton, Reinforcement learning with factored states and actions, J. Mach. Learn. Res. 5 (2004) 1063–1088.
36. S. Lange, M. Riedmiller, Deep auto-encoder neural networks in reinforcement learning, in: The 2010 International Joint Conference on Neural Networks (IJCNN), 2010, pp. 1–8.
37. Nicolas Heess, David Silver, Yee Whye Teh, Actor-critic reinforcement learning with energy-based policies, in: Marc Peter Deisenroth, Csaba Szepesvári, Jan Peters (Eds.), EWRL, J. Mach. Learn. Res. Workshop Conf. Proc. 24 (2012) 43–58.
38. D.H. Hubel, T.N. Wiesel, Shape and arrangement of columns in cat's striate cortex, J. Physiol. 165 (3) (1963) 559–568, http://dx.doi.org/10.1113/jphysiol.1963.sp007079.
39. D.E. Rumelhart, G.E. Hinton, R.J. Williams, Parallel Distributed Processing: Explorations in the Microstructure of Cognition, MIT Press, 1986, pp. 318–362.
40. Long-Ji Lin, Reinforcement Learning for Robots Using Neural Networks, PhD thesis, Pittsburgh, PA, USA, 1992.
41. Razvan Ioan Ionasec, Ingmar Voigt, Bogdan Georgescu, Yang Wang, Helene Houle, Fernando Vega-Higuera, Nassir Navab, Dorin Comaniciu, Patient-specific modeling and quantification of the aortic and mitral valves from 4-D cardiac CT and TEE, IEEE Trans. Med. Imaging 29 (9) (2010) 1636–1651.
42. Xiaoguang Lu, Bogdan Georgescu, Marie-Pierre Jolly, Jens Guehring, Alistair Young, Brett Cowan, Arne Littmann, Dorin Comaniciu, Cardiac anchoring in MRI through context modeling, in: Tianzi Jiang, Nassir Navab, Josien P.W. Pluim, Max A. Viergever (Eds.), MICCAI 2010, Part I, in: Lect. Notes Comput. Sci., vol. 6361, Springer, Heidelberg, 2010, pp. 383–390.
43. Xiaoguang Lu, Marie-Pierre Jolly, Discriminative context modeling using auxiliary markers for LV landmark detection from a single MR image, in: Oscar Camara, Tommaso Mansi, Mihaela Pop, Kawal Rhode, Maxime Sermesant, Alistair Young (Eds.), STACOM 2013, in: Lect. Notes Comput. Sci., vol. 7746, Springer, Heidelberg, 2013, pp. 105–114.