Reinforced Segmentation of Images Containing One Object of Interest

by

Farhang Sahba

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Systems Design Engineering

Waterloo, Ontario, Canada, 2007

Abstract

In many image-processing applications, one object of interest must be segmented. The techniques used for segmentation vary depending on the particular situation and the specifications of the problem at hand. In methods that rely on a learning process, the lack of a sufficient number of training samples is usually an obstacle, especially when the samples need to be manually prepared by an expert. The performance of some other methods may suffer from frequent user interactions to determine the critical segmentation parameters. Also, none of the existing approaches use online (permanent) feedback, from the user, in order to evaluate the generated results. Considering the above factors, a new multi-stage image segmentation system, based on Reinforcement Learning (RL) is introduced as the main contribution of this research. In this system, the RL agent takes specific actions, such as changing the tasks parameters, to modify the quality of the segmented image. The approach starts with a limited number of training samples and improves its performance in the course of time. In this system, the expert knowledge is continuously incorporated to increase the segmentation capabilities of the method. Learning occurs based on interactions with an offline simulation environment, and later online through interactions with the user. The offline mode is performed using a limited number of manually segmented samples, to provide the segmentation agent with basic information about the application domain. After this mode, the agent can choose the appropriate parameter values for different processing tasks, based on its accumulated knowledge. The online mode, consequently, guarantees that the system is continuously training and can increase its accuracy, the more the user works with it. During this mode, the agent captures the user preferences and learns how it must change the segmentation parameters, so that the best result is achieved. By using these two learning modes, the RL agent allows us to optimally recognize the decisive parameters for the entire segmentation process.

Acknowledgements

As I write these lines, I look back over the past years of my life and find myself thankful to several people.

I am grateful for the support and encouragement of my supervisor Professor Hamid Reza Tizhoosh. He taught me to first be a human being and a scientist. He taught me to consider alternative strategies in the research, see the problems from different points of view and helped me to visualize what I want to do. I always learn something new from him. I would also like to express my especial gratitude to my supervisor Professor Magdy M. A. Salama, for his insightful comments and suggestions toward this thesis.

I wish to thank my thesis committee members, Professor Otman Basir, Professor David Clausi and Professor George Freeman and my external examiner Professor Maher Sid-Ahmed, for their thorough reading of this thesis and insightful suggestions.

My deepest gratitude to my wife, Niloufar, for living with this thesis as well as me. This thesis is dedicated to you Niloufar as it would have never been possible without your patience, encouragement and support.

I am also very grateful to my son Faraz for his patience and understanding during the past five years. One of the best experiences of me and my wife throughout this period was the birth of our second son Farbod, who with his brother adds a joyful dimension to our life. Their happy faces remind us how beautiful our life is.

I feel a deep gratitude to my parents, Shamsolsadat Modares and Amir Houshang Sahba, who have sacrificed so much to provide me with the quality of life from which I could choose what I wanted to be. Thank you so much to both of you who have always been a source of spiritual power for me.

I owe immeasurable thanks to my parents in low, Hossein Alamzadeh and Fataneh Mojahed, for their encouragement and continuous support. Thanks for all they have done for me and my family throughout the past twelve years and as well as during the life of this project.

At last, praise to the Almighty God, whose help and guidance has always been present in my life and who gave me the ability and strength to complete this work.

To my wife, my parents and parents in law, and my sons
for their love, encouragement and support

and

To those who had the talent but
never got a chance

Contents

List of Tables

List of Figures

Nomenclature

α	step size
γ	discount factor
Γ_1 , Γ_2	object shape
$\Delta(\tilde{x_1}, \tilde{x_2})$	distance between two strings
Δ	adjustable factor
Δ_g	local gray level step
μ	mean value
ν_c, ν_o	size of structuring element for closing and opening
ω_R	number of rows in sub-images
ω_C	number of columns in sub-images
ω	frequency in Fourier representation
ϕ	relative angle of the sub-image
π	policy
π^*	optimal policy
Ψ	compactness
ρ	relative distance of the sub-image from x_c, y_c
σ	standard deviation
θ	temperature
θ_C	angle assigned to curve points

$\breve{\varrho}$	degree of opposition
τ_i	local threshold value
ξ_i	state components
a	action
a'	next action
a_i	action at iteration i
a^*	optimum action
$\langle a_t \rangle$	sequence of actions starting at t
$\breve{a}$	opposite action
$\mathcal{A}$	set of actions
A_{ratio}	area ratio
A_{object}	area of the object
A_{ref}	area of the reference object
A_{SO}	area of the object located in the sub-image
$B_{\ell,i}$	properties of binary sub-image ℓ in iteration i
$\mathcal{C}$	set of control parameters
$\mathbf{c}_i$	set of control parameters for $Task_i$
$c_{\ell,i}$	control parameters for sub-image ℓ in iteration i
$c(u), x(u), y(u)$	object boundary parameters for Fourier eliptic
$c_{xk}, c_{yk}, a_{xk}, b_{xk}, a_{yk}, b_{yk}$	elliptic Fourier coefficients
C^i	object curves
CE_N	elliptic Fourier coefficients vector
CE_{ref}	Fourier coefficients of reference object

CE_Δ	distance between two Fourier coefficients vectors
D_i	dissimilarity
D_{χ^2}	histogram distance
D_{i_s}	dissimilarity with the desired output
d	distance to the nearest contour point
$f(s,a)$	state transition function
G_ℓ	gray level information for sub-image ℓ
g_{ij}	gray level value
g_{lmax}, g_{lmin}	maximum and minimum local gray level
h_1, h_2	normalized histograms
I	input gray level image
$\hat{I}$	gray level image after preprocessing
I_{OT_i}	output image after $Task_i$
I_{i_s}	output sub-image
$I_{i_{sd}}$	desired output sub-image
i	iteration number
K	maximum number of iterations
ℓ	index of sub-image
L_s	number of sub-images
M	number of training images in offline mode
m_b	number of points on the object contour
N	number of processing tasks
N_F	number of selected elliptic Fourier coefficients
N_{WP}	number of white pixels in the image

N_{TP}	total number of pixels in the image
N_O	number of extracted objects
N_s	number of states
n_R	number of regions in an image
P_ℓ	geometrical characteristics for sub-image ℓ
P_{OS}	perimeter of the object located in the sub-image
$P^a_{ss'}$	probability of transition s to s'
$Q(s,a)$	state-action value function (Q-matrix)
r	reward
$r_{s_i,s_{i+1}}$	reward for transition from state s_i to state s_{i+1}
r^{OFF}	offline reward
r^{ON}	online reward
r^{ON}_{subj}	online subjective reward
R_1,R_2	constant reward values
$R^a_{ss'}$	expected immediate reward on transition from s to s'
R_I, C_I	image row and column numbers
s	state
s'	next state
s_t	state at t
$\breve{s}$	opposite state
$s_{\ell,i}$	state of sub-image ℓ in iteration i
$\mathcal{S}$	state space
$\mathcal{S}^+$	set of all states, including the terminal state
S_{R_i}	regions in the segmented image

T	number of extracted features
t	discrete time step
$V(s)$	value of state s
$V^*(s)$	optimum value function
WP_{ratio}	area ratio of all extracted objects
$\mathcal{X}$	set of features
x_i	the i^{th} feature
$\tilde{x}_i$	point string
x_c, y_c	geometric center of the object
x_p, y_p	coordinates of the pixels on the object border
x_s, y_s	coordinates of the center of the sub-image

Chapter 1

Introduction

When an object or a scene is visually captured by an optical instrument, a digital image can be formed. This visual perception of the real world is the most informative and comprehensive type of data for human perception. In image-based research, the goal is to derive image information from different perspectives, in order to analyze and understand image content. Captured images may not meet the criteria of good quality. Therefore, a sequence of different procedures, called digital image processing, may be needed to perform some sort of enhancements and extract information, such that the image objects can be better identified. These procedures cannot compete with human vision, in terms of recognition capability. But they can outperform it, with respect to the execution of detailed mathematical operations on many images, in a short time.

Image segmentation is one of the most important image processing tasks. It partitions images by determining disjointed and widely homogeneous regions. Image segmentation applications can cover a wide range, including medical diagnosis and treatment, molecular biology, microscopy, remote sensing, geophysical prospecting, space imaging, industrial quality inspection and machine vision. Segmentation areas can range from low-level operations, which use the basic pixel information, to high-level operations including the application of artificial intelligent techniques for object interpretation and analysis. The more recent algorithms, which have been developed to work based on intelligent techniques, usually use a priori knowledge gained from different sources.

This thesis mainly focuses on the introduction of a new system for the segmentation of images containing one object of interest, using an intelligent technique, called reinforcement learning.

1.1 Reinforced Image Segmentation (RIS)

Image segmentation plays a pivotal role in virtually all image-based applications. It can provide the information needed for pattern recognition and/or image analysis [25]. Different image characteristics, such as color and brightness, texture and edginess of image components, may be used for distinct segmentation.

In this study, a multi-stage segmentation system based on reinforcement learning (RL), is introduced. The most important concept of RL is learning by trial and error based on interaction with the environment [77, 79]. This makes the RL agent suitable for dynamic problems. The goal of an RL agent is to find an *action policy* that controls the behaviour of a dynamic process. The agent is guided by signals (reinforcements) that indicate how well it has been performing the required task.

In the case of using RL for image segmentation applications, we encounter a challenging task, which is the application of a dynamic solution to a static problem[1]. In the proposed RIS system (Reinforced Image Segmentation), the agent first takes some actions (i.e.,

[1]A sequence of images may be considered a dynamic environment. However, segmentation of single images is certainly a static one.

changing parameters), to modify its environment (i.e., the quality of the output image) through an offline learning mode. During this stage, which uses manually segmented images, the agent can gain appropriate knowledge for similar images. States are defined based on the information obtained from the final output, as well as intermediate results after each image processing task. We also define actions as reinforced adjustment of the parameters of different processing tasks. A potential obstacle, when we apply RL agents into image-based applications, is the large number of state-action pairs in our problem. In such cases, it is usually difficult to extract the state-action information, especially when we need to store past experience. Opposition-Based Learning (OBL) is one method that can be applied to overcome this problem [85]. This technique offers a faster learning speed in comparison to conventional implementations of RL agents.

Since we wish to apply our method on sub-images, we have to employ states that do not result in extremely large state spaces. In addition, these states must use the features that reflect the quality of the segmented object.

1.2 Motivation

The primary interest of this research was to construct a dynamic segmentation system for images, containing one object of interest, using reinforcement learning. The principle motivation was to overcome problems faced by current knowledge-based segmentation methods, by adding the permanent learning capability to the proposed system.

The first obstacle in learning-based methods is the need for a sufficient number of training samples. Learning-based methods generally rely on preparing many training (ground-truth) samples, which is not a simple job, especially when the sample should be prepared by an expert. In addition, none of the existing methods use online (permanent) feedback from the user, in order to evaluate the result, whereas in many applications an experienced user has to acknowledge the quality of the results at the end of the process. Objective quality measurements provide mathematical evaluation of the segmented images, but they generally fail to capture human visual sensitivities. Subjective quality measures, however, should provide numerical and/or linguistic assessments corresponding to the viewers' satisfaction. Therefore, the main purpose of this work is to design and develop a system for the segmentation of one object of interest that satisfies the following properties:

- Training using a limited number of samples,
- Using a manageable amount of user feedback,
- Acquisition of extra knowledge during online operation.

1.3 Thesis Outline

The remainder of this thesis includes five chapters as follows:

Chapter 2 describes the most common segmentation approaches and briefly reviews low-level operations, which use basic pixel information, as well as those which usually use high-level information to extract the object of interest. The advantages and disadvantages of these methods are described.

Chapter 3 provides an overview of reinforcement learning and opposition-based learning, respectively. The fundamental aspects and components of RL are described in this chapter.

Chapter 4 outlines the proposed Reinforced Image Segmentation (RIS) system. In this chapter, a general framework is introduced to adjust the parameters of a multi-task algorithm for image segmentation, using an RL agent. Detailed descriptions are given with the main goal that a well-structured model should achieve segmentation of one object of interest in two stages, including offline and online modes.

Chapter 5 demonstrates the experimental results for the proposed RIS system. It looks at real images as well as quantitative measures to perform comparative studies.

Chapter 6 reflects on different perspectives of this contribution and highlights the potential for future directions and improvements to this research.

Chapter 2

Image Segmentation

2.1 Overview

By identifying key image components, image segmentation plays an invaluable role in many computer vision applications. This is generally achieved based on some measurable features, such as intensity, texture, gradient, etc. The purpose of image segmentation is to find meaningful regions, that correspond to different real entities. This is achieved by finding the boundaries between the regions. Therefore, in a large number of applications in image processing and computer vision, segmentation is a fundamental step before representation, recognition, and interpretation. The uniformity or homogeneity of segmented regions, with respect to some characteristics such as gray tones, texture properties, and meaningful and spatially accurate boundaries, are among the properties that a good segmentation algorithm should exhibit.

Image segmentation is implemented for two main reasons. First, some images contain a large amount of information. To make this information understandable for automated processing, the image data must be reduced by extracting only the information that best represents the image content. This generally means retaining only the outline of interesting regions in the image to be used for further processing steps. Secondly, some images may contain imperfect information. This usually happens when the boundary between regions cannot be easily distinguished, due to some disturbing factors. In these cases, the image segmentation algorithm must determine the boundaries between the regions as accurately as possible. The demand for robustness, reliability and automation of image segmentation algorithms, has considerably increased in recent years.

An image can consist of a number of objects defined by distinct regions. The image background is also a distinct region by itself. The region boundaries separate them from each other. Image segmentation is the way to identify these boundaries. Each region is separated from others, based on some specific criteria. Due to many factors, such as non-uniform illumination, poor contrast, noise, and imaging artifacts, the process of image segmentation is usually a challenging one. As a result, the objects may not receive very well defined boundaries. In such difficult circumstances, segmentation errors will occur. Under-segmentation and over- segmentation are two major problems that need to be minimized [96].

Under-segmentation is the case when full segmentation has not been achieved and there are two or more regions that appear as one. In over-segmentation, a region is split into two or more parts. Under and over-segmentation are not easy to resolve, but can be minimized based on the application we deal with. The solution to many segmentation problems is highly dependent on the specifications of the corresponding application.

2.2 Segmentation Methods

There is a wide variety of methods available for image segmentation. The main goal of a segmentation algorithm is to partition an image into constituent regions being homogeneous [27, 49]. Image segmentation methods cover a broad range, from basic pixel-based low-level operations to high-level analysis, that use artificial intelligence techniques which usually work based on the domain knowledge, relationship between the components and the history of the image, if it is a part of a sequence. Techniques used in segmentation algorithms vary depending on many factors. Currently, there is no general segmentation algorithm with acceptable results for all applications. But, there are several methods specialized for particular applications. These can usually give better results, by taking expert or a priori knowledge gathered from different sources into account.

Choosing a suitable approach for a specific application is one of the most crucial problems encountered in image segmentation. Image enhancement techniques may be needed to improve the quality of the original image [61]. These techniques highlight the salient features of the original image in order to simplify the task of image segmentation. Choosing appropriate algorithms, measuring their performance and understanding their impact on the next stages, are the issues involved in image segmentation. Performance evaluation is another issue related to segmentation methods. This can be a crucial task, as different parameter settings can significantly affect the final result.

For some segmentation methods, the manual user interaction can be an important issue. Generally, there is a trade-off between user interaction and the overall performance. By incorporating the expert knowledge via manual interaction, the accuracy can be improved. But, in some cases it may be a time-consuming task. More importantly, the user feedback is rarely used to improve the inherent segmentation capabilities of the algorithm. The type of interaction used in segmentation algorithms varies depending on the amount of time and effort required from the user. Generally, many segmentation methods require at least some user interaction to adjust critical initial parameters.

To evaluate the performance of a segmentation method, validation experiments must be conducted. It is typically performed using ground-truth models. The most straightforward method is to compare the result of a segmentation algorithms with manually obtained segmentations [96]. Using a ground-truth model, some criteria can be defined to measure the accuracy. The choice of these criteria depends on the application and can be based on the region information, such as the number of misclassified pixels, or boundary information, such as average distance from the true boundaries [92].

Many methods have been proposed for image segmentation. These methods can be classified according to different perspectives. Some of these possible groupings can be represented as follows [31, 40, 49]:

Region of Interest

- *Boundary-based methods* work based on the detected edges followed by an interpretation of the gradient information.

- *Region-based methods* map individual pixels to pixel sets.

Locality of the Method

- *Local methods* operate on the local parts of an image. They do not use any global image information.

- *Global methods* usually use features of the objects in the image and focus on the whole image instead of local characteristics.

- *Hybrid methods* take the advantage of both local and global image features.

Level of the Method

- *Low-level segmentation* includes pixel-based, localized techniques.
- *High-level segmentation* includes analysis and integration of expert knowledge in a segmentation system.

Basic Elements of Representation

A method for the classification of image segmentation techniques is based on the basic element of representation. According to this classification, the segmentation methods can be divided into the following two techniques.

- *Feature-based techniques* that work based on using certain image features. In these techniques, image segmentation is accomplished by extracting/detecting the homogeneous regions in feature space.
- *Spatial-domain techniques* that take the spatial characteristics of pixels into account. This is based on the important fact that points of an object are usually spatially close, due to surface coherence [31].

Without attempting to choose or introduce any specific categorization, the most commonly used segmentation methods will be briefly described in the following subsections.

2.2.1 Histogram Thresholding

Histogram thresholding segments images by creating partitions. It is one of the most practical methods and can be quite effective for some image classes [49, 70]. The histogram is based on some global characteristics of the image, such as brightness distribution. In a general form for an image I, the basic idea of histogram thresholding is that if we are looking for

n_R regions, we are seeking to find $n_R - 1$ values $\tau_1, \tau_2, ..., \tau_{n-1}$ in the range of the histogram, which unambiguously separate those regions. This is known as a multi-thresholding approach and specifies a range of pixel intensities that belong to the foreground.

When there is a clear difference between the gray levels of the objects (e.g., image has a bimodal histogram), selecting a threshold is trivial. Bi-level image thresholding separates the related areas in the image into two categories and presents those by using two different intensities black and white. This method has been applied to simple cases, typically containing two classes: object and background [13, 55]. But, when the image has non-uniform illumination and the background intensity varies across the image, or when it contains noise, the histogram has an irregular shape. A common problem in these cases is that the profile of the histogram is more jagged, which gives rise to many peaks and creates segmentation ambiguities. In some cases preprocessing techniques can be adopted to prevent this from happening. Measures of fuzziness, such as fuzzy entropy can also be used in thresholding tasks [87].

To make thresholding an applicable method for the segmentation of complex images, it is needed to apply it locally to take the benefits of an adaptive implementation, which incorporates information based on local intensities. As the pixels that are segmented as an object are not necessarily connected some post-processing operations may be required.

There are several threshold selection methods introduced in literature that can be divided into specific categories, based on the information they are using [68, 70].

2.2.2 Classification and Clustering-Based Segmentation

Clustering is an unsupervised classification of objects to generate different partitions, usually without using a priori knowledge. In the case of image segmentation, the process of clustering generates some regions $S_{R_1}, ..., S_{R_n}$, such that every pixel g_{ij} belongs to one of these regions and no pixel belongs to two regions at the same time. Classification techniques partition images using data with known labels. These techniques follow the general idea that the objects within each class should show a high degree of similarity, while across different classes they should exhibit very low similarity [71]. Classifiers require training data and therefore are generally known as supervised techniques. In fact, they use the manually segmented images as training samples and subsequently use the acquired knowledge to automatically segment the new data. Many techniques have been proposed in literature for cluster analysis [5, 11]. For instance, the nearest-neighbour classifier can be used as a simple method. In this case, pixels are classified in the same class based on the intensity distance. The k-nearest-neighbour classifier is a generalized version of this approach in which pixels are classified into the same class as the majority of the k-closest training data. A commonly used parametric classifier is the maximum-likelihood or Bayes classifier [82]. It assumes that the pixel intensities are independent and can be modeled from a mixture of probability distributions, which are usually assumed to be Gaussian. Fuzzy image processing provides some new segmentation techniques [83]. Fuzzy clustering is the oldest fuzzy approach to image segmentation. In this technique, the clusters are built as segmented objects using fuzzy c-means and possibilistic c-means. The image features can be interpreted as linguistic variables, and then use fuzzy if-then rules to segment it into different regions. The weakness of these classifiers is that they usually do not perform any spatial modeling. This may cause a problem in images that are corrupted by intensity inhomogeneities [33]. Clustering analysis can be performed either by providing a seed point for the regions to be segmented or using non-parametric methods for finding the salient regions without a seed point. Among clustering algorithms, three of them are most commonly used. They are the k-means algorithm [82], the fuzzy c-means algorithm [38], and the expectation-maximization (EM) algorithm [9]. Some issues related to clustering include how many clusters are needed or how to determine the validity of clusters. In some techniques, such as fuzzy c-means clustering, the number of clusters in the image must be specified in advance. Some other techniques do not require this initialization [53]. The weakness of these algorithms is that they are sensitive to initialization. They do not use spatial modeling directly and can therefore also be sensitive to noise and intensity inhomogeneities. As an example for medical applications, in the segmentation of a slice of a Magnetic Resonance (MR) brain image consisting of three classes that represent dark gray (cerebrospinal fluid), gray matter, and white matter, the k-means algorithm can be applied [39]. In this case, the robustness of clustering algorithms in treating intensity inhomogeneities is the main concern. The traditional clustering techniques assign pixels to clusters only on the basis of their gray level values and no spatial constraints are considered. But, there is a special classification devoted to a category of segmentation algorithms that combines the idea of k-means clustering with the desirable local properties of gray levels. This class of algorithms that use spatial continuity lie between the feature-based techniques and the spatial-domain techniques [50]. In some cases, such as the k-means algorithm under a Bayesian model, Markov Random Fields (MRFs) are incorporated into clustering segmentation algorithms [28].

2.2.3 Region Growing

Region growing is one of the most popular segmentation methods for a large number of image classes. This method relies on the homogeneity of the image regions using some predefined criteria, such as color, gray levels and texture [3]. Region growing algorithms take one or more pixels, called seeds, and gradually enlarge the regions around them to satisfy a certain homogeneity criterion. For instance, one possible criterion can be meeting an edge before growing is stopped. If the adjacent pixels are similar to the seed, then they are merged within a single region. The growth process stops when no more points can be added to the region and all pixels in the image are assigned to at least one region. The techniques based on region growing mainly aim at processing single regions. By combining different subsequent growth processes, however, we can force the regions to include all pixels. In these cases, some very small regions may exist or some neighbour regions grown at different times, which exhibit similar characteristics. A merging phase, as a post-processing stage, can be used to generate meaningful border regions. Methods using this approach employ splitting the image into a large number of small regions, and recursively merging smaller regions into larger regions using some suitable criteria. Among the methods used in this category, merging by the nearest neighbour or the k-nearest neighbour rules, boundary melting and splitting can be listed [94].

The main advantage of the methods based on region growing is that the regions obtained are spatially connected. A disadvantage of the region growing techniques is choosing suitable seed points and the existence of an adequate homogeneity criterion. Another weakness of region growing methods is that they are usually sensitive to noise, which causes holes or discontinuities in extracted regions. Region growing is usually used with a set of image-processing operations. It can be used for the delineation of simple and small structures such as tumors and other lesions [23, 54]. To improve its performance, homotopic region growing algorithms and fuzzy analogies to region growing have also been proposed [41, 87].

2.2.4 Split-and-Merge Techniques

Algorithms based on split-and-merge, are methods related to region growing with some differences [42]. The techniques based on split-and-merge usually start with an initial inhomogeneous partition of the image. This partition can be the image itself. They continue to split the partition until homogeneous partitions are obtained. As a multi-resolution scheme, quadtree representation is a common structure used to implement this procedure [31]. After this phase, there usually exist many small and fragmented partitions that have to be connected in some way. The merging phase implements this task by associating neighbouring regions such that the homogeneity requirements are met up until a maximally connected segment can be reached. The Region Adjacency Graph (RAG) is the data structure commonly adopted for merging phase [11].

In some algorithms, the smoothness and continuity of partitions are enforced using a Markov Random Field (MRF) [22]. This contains the property that the conditional probability of taking a certain value by a particular pixel is only a function of the neighbouring pixels, not of the entire image.

2.2.5 Template Matching

Sometimes it is needed to know whether the image includes some particular objects. Template matching is a classical technique for this purpose. It is the measure of how well a portion of an image matches a specific template. Euclidian distance is usually used to measure the difference. Statistical interpretations of this method can also be employed using prior probabilities of the objects in the image. Another way is using the correlation function approach for template matching. The idea of template matching generally does not take the effects of rotation and scaling of the object into account. Therefore, different templates have to be applied for the detection of the rotation and/or scaling, which increases the complexity of the method significantly.

A specific class of template matching techniques is *model-guided segmentation*. This technique uses library models to segment images. It uses the prior information about the shape of interest, to segment the objects with little user interaction. This technique first finds a one-to-one transformation that maps a pre-segmented model to a new image. It is often known as model warping. Some sequential applications of linear and nonlinear transformations may be needed as well [16, 69]. This approach can be used for problems where the segmented training models are available and must be matched with the input images. Usually, when a standard atlas or model is available and the input images are highly correlated, this technique works well. For instance, in the field of medical imaging, a standard atlas is generated using the analysis of information on the anatomy that requires segmentation. This standard atlas is then used as a reference frame to segment new input images. The approaches that work based on template matching are similar to classifiers except that they are implemented in the spatial domain. An advantage of the atlas-guided approach is that the labels are transferred with the segmentation procedure. However, finding accurate segments in complex structures is usually difficult. This technique has been used for the segmentation of various structures in MR brain imaging [16] and for extraction of the brain volume from head scans [2]. Some disciplines, like anthropometry, physical anthropology, neuroscience, oncology, and prosthetics use this technique to extract the desired image segment.

2.2.6 Morphological Watershed Segmentation

Mathematical morphology, built on set theory, has been used in many applications in image processing. It is based on the generation of mappings for each pixel in a local neighbourhood. Some previous works have used morphology techniques to segment an object in the image. It includes various operations such as erosion, dilation, opening and closing, which can be used in image segmentation tasks. A well-known morphological segmentation is the watershed algorithm, which uses concepts of edge detection to partition images into homogeneous regions [88]. In some cases, it may suffer from over-segmentation where the image is segmented to an unnecessarily large number of regions. To overcome this problem, the segmentation is usually followed by a post-processing step for merging separate regions that belong to the same structure [76].

2.2.7 Edge-Based Segmentation

Edge-based segmentation methods involve finding the edges of the objects and using them to find complete boundaries. Principal edge detection techniques are usually convolution-based operators. This family of operators typically detects gradients in a number of directions and combines these results to produce an edge map. Generally, convolution operators are defined as 2×2, 3×3 or 5×5 sliding windows, which are convolved with the image. These results are then combined to create the final edge map. The simplest of these filters is the Roberts operator. Among many other operators in this category, the most popular operators are Prewitt, Sobel and the Laplacian gradient operator [25]. The Sobel and Prewitt operator approximate the first derivative in a specific direction, and the Laplacian operator approximates the second derivative of the image. This family of filters has acceptable performance for images with unambiguous edges. However, applied on more textured images, or on images with noise and sub-optimal contrast, they tend to produce false edges, or fragment the true edges [25]. Therefore, some post-processing techniques are required to refine the results obtained by edge detectors. Edge relaxation and border tracing are among alternatives for this purpose. Edge relaxation techniques attempt to solve the two problems of removing false edges and completing edges which have lost continuity. They usually work well, but are computationally expensive. After having a complete edge map, the task is to isolate the regions in the image for segmentation. One of the popular and most intuitive methods for doing this is using border/contour tracing in medical image segmentation [78].

2.2.8 Deformable Models

Deformable models are promising model-based segmentation approaches. They have been widely used and usually require expert interaction. Deformable models are capable of coping with significant variability of structures. Their potential for use in medical image analysis was quickly realized. Deformable models (or "active contours") are basically derived from the concept of the snake model introduced by Terzopoulos and Kass [34]. The snake model follows the edges in an image by considering a curve inside or outside of the object, and then let the curve expand/shrink to a suitable shape and position. This curve has some physical properties, such as elasticity and rigidity. In these approaches, an energy-minimizing contour, called a "snake", is controlled by a combination of three forces:

(i) internal contour force, which enforces the smoothness,

(ii) image force, which attracts the contour to the desired features, and

(iii) external constraint force.

Each force creates its own potential field; the contour actively adjusts its position and shape until it reaches a local minimum of the potential energy. When an appropriate initialization of the contour is specified, the snake can converge to its minimum energy. The deformable model is a powerful interactive tool for medical image segmentation. However, because of the strict use of local image information, this approach may be trapped in the local minimums. The implementation of the original model is highly dependent on the image noise and the initial snake position. Some attempts have been made to improve the robustness and stability of these models. For instance, an approach is introduced as "balloon force", which can inflate or deflate the contour [15]. This force helps the snake to pass false, isolated and weak edges, and improve its tendency to shrink/expand. The resulting model is more robust with respect to the initial position and image noise. The main advantage of deformable models is the ability to directly generate closed parametric curves from images. The disadvantage is that they require manual interaction for placing an initial model and choosing appropriate parameters. Also, in deformable models, energy minimization is generally computationally expensive, and achieving the local minimum is always a problem. The latter is especially true when the image is corrupted by noise. Snakes have been used to locate structures in the brain, retina, vertebra, neuronal tissue and other parts of human body [34]. For instance, one area in which they have often been used, is the reconstruction of the cerebral cortex from MR images, or in MR heart images [20]. Deformable models have also been applied for the segmentation of cardiac images [7], bone in CT images [45] and echocardiographic sequences in ultrasound images [44].

2.2.9 Artificial Neural Networks

Artificial neural networks (ANNs) can be used in different ways for image segmentation [26]. In these methods, weights are trained using sample data and the network is then used to segment new image data. ANNs offer high degree of parallelism, which allows for very short computational times. Due to the structure of neural networks, spatial information can be incorporated into the classification procedures. In general, pixel-based ANNs are trained to classify the image content based on texture or a combination of texture and local object shape. A number of algorithms have been proposed for the segmenting of gray level images with ANNs. Artificial neural networks have also been developed for pre- and post-processing steps in the segmentation procedure, such as connecting edge pixels, delineation of contours or deciding whether a pixel lays inside or outside a segmented object [26]. Also, a number of pixel-based approaches have been proposed that apply ANNs for segmentation by direct clustering of the pixels. There are some other ANN segmentation techniques that work based on the features, including color, position, size, orientation, shape (using principal components), texture (using co-occurrence matrices; wavelet features or multi-resolution features extracted from the Gabor wavelets) and a combination of texture and local shape. Besides direct classification, ANNs have been used for connecting edges and lines, or in region growing techniques [36].

2.3 Bottom-Up Versus Top-Down Segmentation

One strategy to segment images, called the bottom-up approach, first segments the image into regions and then recognizes those corresponding to a particular object. This strategy groups the pixels based on their gray levels, texture information, smoothness and continuity of object boundary.

The segmentation approaches based on the bottom-up technique use different criteria and search for homogenous segments within the image. For instance, a way to use the bottom-up approach is to represent the image as a graph such that the nodes correspond to pixels and then segment the graph into subsets corresponding to relevant regions. The features for segmentation can be texture, intensity and boundary properties of different regions. In this representation, the bottom-up approach applies segmentation on the homogenous regions at a given level such that they make larger homogenous regions at the next level. This produces a weighted structure, such that each region is connected to other regions in the next level with a relating weight. The main difficulty of the bottom-up approach is that some of the produced partitions may wrongly merge to the background.

Another approach, called top-down segmentation, uses prior knowledge of the object, such as its shape, texture, or colour to guide the segmentation. The top-down approaches rely on acquired class-specific information and can be applied to images from a specific category. They may deal with the variability of shape and appearance within a category. If there is a large variability in the shape and appearance of objects, the top-down approach may face difficulties and the segmentation results may not accurately match the object boundary.

2.4 Image Segmentation Evaluation

We have discussed several methods for image segmentation. It is known that no method can be applied to all images and not every method is suitable for a particular type of image. Therefore, the important issue is how to perform a quantitative evaluation of segmentation results. This is to compare different algorithms or, in general, to assess the quality of extracted segments. Such a quantitative measure is useful not only to justify an algorithm, but also for vision applications, where automatic decisions are required for the stages after the segmentation. As a matter of fact, this often turns out to be a difficult task. The reason for this difficulty is that the quality measures are highly related to the desired segmented image, which is generally subjective. The desired segment can be generated either manually or via a reliable procedure in order to compare the results of the algorithms. These reference segments are called *ground-truth* or *gold standard*. Ground-truth images, however, are observer-dependent in their representation of the perfect output, because two different persons may perform different manual segmentation on the same image. Because no other independent knowledge of the solution may be available, manual segmentation is nevertheless the best procedure to evaluate the output of any segmentation algorithm. Although, it generally gives the best and most reliable results for many tasks, it is not easily applicable in all cases and therefore some attempts have been made for the quantitative (objective) evaluation. Evaluation of the image segmentation method is based on the efficiency of the applied method in terms of time, number of parameters needed to obtain a solution, statistical measures of performance including accuracy, and so on.

2.4.1 Subjective Quality Assessment

The subjective quality evaluation consists of the assessments by a group of human viewers. Subjective quality measures, like MOS (Mean Opinion Score), provide numerical values that quantify viewers' satisfaction with processed images. These methods involve observer-oriented procedures based on the models of human visual perception. In particular, when artifacts are close to visual thresholds, the perceptual measurements can provide more consistent calculations of image quality in comparison to objective measurements. Implementation of these models is generally accompanied by complex and time-consuming subjective psychophysical testing for validation [21].

The observer is a key aspect of the image quality evaluation in many environments. Receiver operating characteristic (ROC) analysis has been used to measure the quality of processed images. The ROC analysis is based on the observer rating for each image, which represents his/her subjective assessment of quality [80]. ROC methods are appropriate for tasks with binary "good/not good" decisions or other tasks involving some specific decisions about the quality.

2.4.2 Objective Quality Assessment

Objective quality measurements provide mathematical tools for the comparative evaluation of processed images. Several metrics have been proposed to objectively evaluate the performance of segmentation techniques [12, 18]. Depending on the availability of a ground-truth, evaluation metrics are divided in two categories as follows:

I) Stand-alone evaluation metrics

If a ground-truth is not available, an objective evaluation should rely on available a priori knowledge about expected properties of the objects of interest [18]. The stand-alone evaluation of individual objects can be computed based on the features of the objects themselves (intra-object homogeneity features) or using the comparison of them with those of neighbouring objects (inter-object disparity features).

I.1) Intra-object homogeneity features

These features indicate the internal homogeneity of each object. According to the segmentation application, it can be measured using the following spatial features:

- Shape regularity features

  These types of features are applicable when the objects are expected to have regular shapes which can be evaluated by some geometric features.

- Spatial uniformity features

  In those cases where the texture or gray level of the objects is expected to be uniform, features such as texture or gray level variance can be used to measure the spatial uniformity.

I.2) Inter-object disparity features

The features in this group indicate whether the objects were correctly identified among their neighbours. The object features can be compared to those of its neighbours to give useful information for stand-alone evaluation. The main assumption here is that the objects are sufficiently different from their neighbours. The feature comparison can be locally implemented along the object boundaries, or based on features obtained from the all objects. The following features can be used in this regard:

- Local contrast relative to neighbours

  If there is a significant difference between the inside and outside of an object, the local contrast to its neighbours can provide useful information.

- Neighbouring objects feature difference

  The feature related to the object area can be compared with those from its neighbours to ensure that they have been correctly separated. Shape regularity and spatial uniformity in the corresponding objects are among these features.

II) Relative evaluation metrics

This metric is used when a reference segmented image, as ground-truth, is available. In this case, substantially more information exists than with a stand-alone evaluation, and more reliable evaluation can be obtained. Relative evaluation metrics use information of the reference segments and compare them to the results of the segmentation method. The objective evaluation can be either applied to each individual object or to the overall partition recognized by the segmentation algorithm. In the individual case, both object qualities and the match between the ground-truth and estimated objects and their relevancies are taken into account.

While individual object quality evaluation is needed to assess the output for the subsequent stages, such as pattern recognition, overall segmentation quality evaluation can be used to determine whether segmentation goals have been met.

II.1) Individual object quality evaluation

The algorithms used for the objective segmentation evaluation of each individual object include spatial and temporal features as follows:

- Spatial accuracy

Ideally, the outputs of a segmentation method must be very similar to those in the ground-truth image. When this is not the case, however, a set of related features must be compared such that more spatial segmentation errors return lower quality values. The spatial features used, can be listed as follows:

Shape fidelity - The detected edges are often evaluated by measuring the shape fidelity. For this purpose the percentage of misclassified edge pixels and their distances to the reference object edge is calculated.

Geometrical similarity - It is computed based on the similarities of the geometrical features such as size, position, perimeter and circularity, between the segmented object and the reference object.

Edge content similarity - Using the edge content, the similarity of the segmented object area in terms of spatial complexity with the reference object can be measured.

Statistical data similarity - This measure returns the similarity of some statistical properties, such as brightness, between the segmented object and the reference object.

- Temporal accuracy

  These measures are used for video streams and evaluate the fidelity between the motion in the reference object and the segmented object. Also, there are some other metrics, called spatio-temporal accuracy, that measure spatial and temporal characteristics simultaneously.

Relative segmentation quality evaluation for each individual object, follows three main steps [17]:

1. Segmentation - The segmentation algorithm is applied to the test images.

2. Object selection - The object of interest is selected.

3. Evaluation - The objective evaluation algorithms are applied to measure the segmentation quality.

II.2) Overall segmentation quality evaluation

Quality evaluation for overall segmentation, is performed in four main steps:

1. Individual object quality- Based on the former procedure for each object, the individual object segmentation quality is calculated.

2. Object relevance- The relevance of each object in the image is computed. This is calculated based on how much human visual attention it captures.

3. Similarity of objects- A measure of how well the objects in the ground-truth match with the estimated objects is calculated.

4. Overall segmentation quality evaluation- Based on the individual quality evaluation of the objects, their relative importance for the whole image and their similarities to their reference, the overall segmentation quality metric is obtained.

As a matter of fact, a huge gap still exists between the subjective (expert) and objective (mathematical) approach to image quality assessment that makes fully automated image segmentation an impossible task in many applications.

2.5 Chapter Summary

Image segmentation is an important task in identifying objects of interest in image analysis. The purpose of image segmentation is to provide the information required for other procedures, such as pattern recognition and classification. A general overview of segmentation techniques as well as segmentation evaluation was provided in this chapter. There are numerous techniques for the segmentation of various image classes. They can be categorized in different ways.

Static methods such as snake have static performance. They can generate good results for specific applications only. In addition, they do not have any ability to be trained. It means that a specific procedure such as initialization or parameter setting must be done for each case.

In the methods which use learning algorithms, we usually need a large number of training samples. Moreover, they do not have online training unless we “retrain” them. Also generalization is a common problem in most of them. They can perform well for those cases they have already seen, but they usually do not have a good performance for “unseen” cases.

To the best of author’s knowledge, none of the existing segmentation methods can start their work using a limited number of training samples. In addition, most of them require user intervention to determine critical parameters. Furthermore, none of them can improve their performance using online training.

The next chapter explains an intelligent technique, called reinforcement learning, which enables us to design a segmentation system, with the potential of having the above-mentioned desired properties.

Chapter 3

Reinforcement Learning and Image-Based Applications

3.1 Overview

Learning is the pivotal aspect of any intelligent system. The main idea of Reinforcement Learning (RL) draws from the fields of psychology, neuroscience, statistics, and computer science. It is based on the idea that an agent learns proper behaviour through trial-and-error interactions with a dynamic environment [79].

The RL agent automatically determines the ideal behaviour, within a specific context, that maximizes performance with respect to some predefined measures. It works very differently compared to supervised learning and does not need a set of training examples. Instead, it can continuously learn and adapt while performing the required tasks. This is one of the advantages of RL over other machine intelligence schemes. This can be a very useful trait for those cases when learning data is difficult or impossible to gather. The design of the RL agent is based on the characteristics of the problem at hand. The

RL agent learns from its own experiences. This means it exploits the knowledge it has previously gained through past exploration. A careful balance between exploration (trying new actions) and exploitation (taking rewarding actions), is very important to the success of the RL agent.

Several components are contained in reinforcement learning. As a decision maker, the agent attempts to take an action that influences the environment. It then receives a reward or punishment from its environment depending on the action it has taken. It also receives information concerning the state of the environment (e.g., attributes or features describing the environment) and eventually learns to take those actions which are the most rewarding. During the learning process, it attempts to meet a certain goal related to the state of the environment. It makes the RL agent more suitable in situations where expert knowledge is not available [77]. Reinforcement learning has grown in recent decades, with applications ranging from a backgammon-playing game to space exploration, automated planning, remote systems, planet surface exploration, stability control in power systems and so on. The majority of existing works concerning image-related tasks and reinforcement learning are related to robotics.

3.2 General Concepts

The main idea of reinforcement learning goes back to optimal control problems. Value functions and dynamic programming were introduced as possible solutions for this problem. The goal is to design a controller to minimize a function value over time. Richard Bellman introduced a solution to this problem [79]. The policy is described as any rule for making decisions. Optimal policy maximizes a function of the final state variables.

As learning is based on reward and punishment, reinforcement learning could be considered as semi-supervised learning. Objective rewards and punishments are defined based on some general characteristics of the results. In subjective cases they are provided by an experienced operator. He/she observes the results and gives the expected rewards or punishments. The RL agent is an interactive and goal-oriented learning entity. It can sense its environment and choose the action to change it [79].

The RL agent's behaviour is determined by its own experience [59]. It attempts to map situations (as states) to actions to maximize a numerical reward function indicating the goal of an RL problem [57].

One of the challenges in designing reinforcement learning is how exploration could change to exploitation. The "exploration" occurs when the agent begins learning and does not have any knowledge about how well an action performs. The agent tries to take different actions and observes the rewards, to find out which action yields higher rewards. After a while the agent does have sufficient knowledge about actions. The agent then goes for "exploitation" of what it has learned. It tries to take those actions, which bring more rewards. A tradeoff between the exploration of unseen states and the exploitation of familiar (reward-bringing) states is crucial for RL success. The transition is usually gradual.

A general model of reinforcement learning is illustrated in Figure 3.1. In this model the process is performed as follows [57]:

- Agent observes the states of the environment
- Agent takes an action and observes reward and punishment
- The state is updated and agent observes the new state

The model of the environment is based on its dynamic behaviour. The states are features

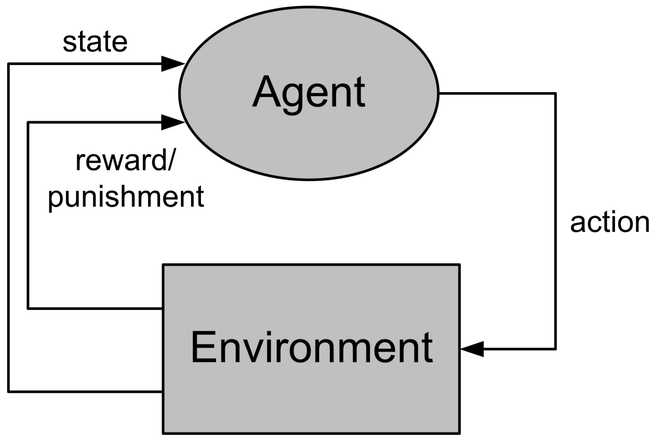

Figure 3.1: Standard model of reinforcement learning.

that describe the environment. The agent must take various actions to change these states and finally optimize them to reach a final or steady state.

In the immediate sense, a reward function assesses the result quality, but a value function implies what is good in the long run [74]. The value of a state is the amount of accumulated reward that the agent expects to gather over time. The reward function maps the state-action pairs to a single number as accumulated reward.

Another important element of reinforcement learning is the action policy that defines the agent's behaviour at a given time. It maps the visited states to proper actions. There are three common policies, namely Boltzman, ϵ-greedy, and greedy. The Boltzman policy is a softmax method using a Gibbs distribution that estimates the probability of taking an action a in a given state s [79]:

$$P(a) = \frac{e^{\frac{Q(s,a)}{\theta}}}{\sum e^{\frac{Q(s,:)}{\theta}}}, \tag{3.1}$$

where $P(a)$ is the action probability. The temperature θ is initialized with a high value and will decrease when the number of iterations increases [79]. In this equation, Q(s, a) is extracted from the Q-matrix, which represents the state-action values (accumulated rewards). In the greedy policy, the agent always selects greedy actions. In this situation, all actions will not be explored. In the ϵ-greedy policy the agent selects greedy actions (which have the highest Q-value in a given state) with probability of ϵ and selects a random action with a probability of $1-\epsilon$. The ϵ-greedy is a popular method to balance exploration and exploitation. Generally, choosing the appropriate policy depends on the application at hand.

3.2.1 Markov decision processes

The concept of Markov decision processes (MDPs) facilitates the understanding of reinforcement learning to a great extent [77]. MDPs are the most convenient way of representation to model stochastic domains, such that optimal behaviour can be estimated. An MDP has the four following elements:

- a set of states $\mathcal{S} = \{s_1, s_2, ...\}$;
- a set of actions $\mathcal{A} = \{a_1, a_2, ...\}$;
- a reward function $r(s, a)$; and
- the state transition function P, which specifies the probability of the next state of the environment, considering the current state and action.

MDPs assume that the next state dynamics are widely independent of previous environment states, rewards or actions. This means that, in order to predict the next state s_{t+1}, all the required information is provided in a compact representation by the current state s_t. This can be formulated as follows:

$$P(s_{t+1} = s', r_{t+1} = r | s_t, a_t, r_t, ..., s_0, a_0) = P(s_{t+1} = s', r_{t+1} = r | s_t, a_t). \tag{3.2}$$

A stationary policy $\pi(s) : \mathcal{S} \rightarrow \mathcal{A}$ for an MDP is a mapping π from $\mathcal{S}$ to $\mathcal{A}$, where $\pi(s)$ is the action that an agent takes in state s.

As stated, the goal is to estimate the optimal value function for a given Markov decision process. Three methods have been proposed for solving this problem. These methods are Dynamic Programming (DP), Monte Carlo (MC) methods, and a third family of algorithms, Temporal-Difference (TD) methods that can be regarded as a link between the DP and MC methods. These three methods are explained in detail in Appendix A.

3.2.2 Q-Learning

Q-learning is an off-policy TD control. This means that while the agent typically follows a policy based on exploitation, it occasionally selects the actions that are suboptimal according to Q value. Q-learning is one of the most popular methods in reinforcement learning, introduced by Watkins in 1989 [90]. In this model-free RL, the agent learns to act optimally in Markovian domains by experiencing sequences of actions. It takes an action in a particular state and uses immediate reward and punishment to estimate the value of that state. By trying different actions in different states, the agent learns which action is best for which state, as judged by the long-term discount reward [91]. The agent must determine an optimal policy π^* and maximize the total discounted expected reward. If we define state s_t, and action a_t at each time step t, we have

$$s_{t+1} = f(s_t, a_t), \tag{3.3}$$

where s_{t+1} denotes the system state at time $t + 1$. In this definition $s_t \in \mathcal{S}$, $a_t \in \mathcal{A}$ where $\mathcal{S}$ and $\mathcal{A}$ are finite sets of possible states and actions $\forall t \geq 0$. One step later, the agent receives a numerical reward $r(s, a)$, and finds itself in the next state s_{t+1}. The goal is to find a sequence of actions $\langle a_t \rangle = a_t, a_{t+1}, a_{t+2}, ...$ applied to the system so that for each possible starting state s_t the following discounted expected reward is maximized:

$$R(s_t, \langle a_t \rangle) = \sum_{k=0}^{\infty} \gamma^k r(s_{t+k+1}, a_{t+k+1}). \tag{3.4}$$

Here $\gamma \in [0, 1]$ is the discount factor.

The action policy is defined as a learning strategy to behave optimally at each time. The optimal state value is the discounted expected reward that the agent will gain if it starts in that state and follows the optimal policy π^*. The corresponding state values are defined as

$$V^*(s) = \max_{\pi} R(s, \langle a_t \rangle). \tag{3.5}$$

As it can be seen, the value function specifies what is good in the long run. Starting from a state, it indicates the total amount of reward that an agent can expect to accumulate. One can show that this equation can be expressed as follows [32]:

$$V^*(s) = \max_{a \in \mathcal{A}} [r(s, a) + \gamma V(f(s, a))]. \tag{3.6}$$

Therefore, the optimal action can be given as

$$a^*(s) = \arg\max_{a \in \mathcal{A}} [r(s, a) + \gamma V(f(s, a))]. \tag{3.7}$$

The action value function for policy π can be given as

$$Q(s, a) = r(s, a) + \gamma V(f(s, a)). \tag{3.8}$$

By substituting the equation 3.8 in equations 3.6 and 3.7 we will receive:

$$V^*(s) = \max_{a \in \mathcal{A}} Q(s, a), \tag{3.9}$$

and

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

According to Watkin's method, let $Q(s, a)$ be the expected return or action-value function for taking action a in state s and continuing thereafter with the optimal policy. The Q-values are estimated in an interacting procedure between the RL agent and the environment. This finally provides a straightforward method to determine the optimal actions for any possible initial state. The stages of Q-learning are presented in Figure 3.2 where s is state, a is action, and s' is next state.

```
Initialize $Q(s, a)$ arbitrary
Repeat (for each episode):
        Initialize $s$
        Repeat (for each step of episode):
        Choose $a$ from $s$ using policy derived from $Q$ (e.g., $\epsilon$-greedy)
            Take action $a$, observe $r, s^{'}$
            $Q(s, a) \leftarrow Q(s, a) + \alpha[r + \gamma \max_{a^{'}} Q(s^{'}, a^{'}) - Q(s, a)]$
            $s \leftarrow s^{'}$;
until $s$ is terminal
```

Figure 3.2: Q-Learning: An off-policy TD control algorithm [91].

3.3 RL and Image-Based Problems

Image-based tasks are procedures that process the visual information. This information can be in the form of the image itself or presented as an image. In each case, the data can be analyzed and manipulated in many ways. To identify the best parameter set in an image-based application, one should try all possible parameter combinations, evaluate them and select the best set. However, this is, in most cases, not possible due to high dimensionality of the parameter set. In this regard, the systems that separate the image analysis procedures and representation of knowledge have been used in various fields such as robotics, medical imaging, remote sensing, and manufacturing. There are various types of knowledge that can be used. In fact, most algorithms must be optimized for specific applications, where the parameters are set for the complete process.

In the case of applying RL to image-based problems, the state of the environment is a major component. For these problems, it is natural for the agent to incorporate image information into its states [60, 64, 65, 66, 67]. Generating states directly from image information results in massive states. It causes difficulties especially in the case of high resolution images that cannot be handled even with value function approximation techniques. Further difficulties arise when the images contain noise and irrelevant data, which create further complications in state generation. More compact states can be generated by incorporating image features instead of raw image information. The concept of policy is used as a mapping from states to actions. The action is defined as changing the parameters that affect the output image. These parameters belong to the algorithms that manipulate digital images in some way to achieve the desired output. Algorithms may range from simple operations to those having more complexity. Another issue is the definition of reward. The RL agent learns only through this signal received from its environment. Therefore, a reward that accurately represents the goal of the agent is crucial to successful learning. The reward must be a single value quantifying the success of learning procedure for a particular goal [81].

3.3.1 Related Studies

There are some image-based applications that have used reinforcement learning. Robotics has used RL for problem solving, resulting in the greatest number of published works in this area. In the field of robotics, the approaches have actively used the direct-vision-based RL, where robots are directly provided with raw visual signals. Usually these signals are forwarded directly to a neural network, with or without preprocessing. These studies have usually used very low-resolution vision systems and demonstrated success in applying their frameworks to real robots seeking targets in an obstacle-navigation task [4, 6, 29, 73]. There are some other studies, which are traditionally classified as image processing applications. For instance, a neural network is trained by RL to classify machine parts in low-resolution images [48]. In this work an extension to higher resolution images by increasing the network size is proposed as well. In another approach, RL-based image thresholding is introduced where entropy is used as a reinforcement signal. The results are compared with an existing genetic method and shown to be superior in quality and computational expense [93].

The application of Q-learning to global thresholding is introduced in [74, 75]. The implementation is one-step episodic. The approach is different from the work in [93], but results are comparable. Two methods are presented in [81, 86], to determine the membership functions in contrast adaptation and parameter control for text detection. The aim of the first work is to examine whether or not RL agents could learn to choose one of a discrete set of simple gray level linear point transformation functions given a degraded input image. State information is obtained using histogram statistics, and reward is a normalized Mean Opinion Score (MOS). In the second work, parameters of the text detection are optimized using RL. Generalization for state space is performed with the fuzzy ARTMAP. A method based on a combination of closed-loop object recognition and segmentation is introduced in [52]. This system uses a reinforcement signal for a team of learning automata to search for segmentation parameters during training. This approach uses some global information and makes adjustments for segmentation. Also, its time complexity is high for real-world applications. In [10] the authors introduce an approach for object recognition that can adjust the parameters in a single stage system. The parameters are represented by a team of generalized learning automata and learned using a connectionist technique. The method uses a model matching signal as a reinforcement feedback, to provide evaluation in a closed loop object recognition system. It alternately switches between global and local processes to optimize the recognition performance. The method has been used for the segmentation and recognition of indoor and outdoor colour images where the important assumption is that the area of the object is known. The only change is the variation of lighting conditions, while the position, shape, and size of the object in all test images are almost the same.

As can be seen, none of the above works have used RL for segmentation of an image, using parameter adjustment in a general multi-task framework. The performance of a few of these works, that are related to segmentation, are limited to very specific conditions in the input images. Moreover, none of them can perform online learning to improve their performance.

3.4 Opposition-Based Reinforcement Learning

Opposition-Based Learning (OBL) provides a practical scheme for the extension of existing learning algorithms (See Appendix B). The idea can be employed to extend RL agents to shorten the exploration time. Tizhoosh [84, 85] has introduced a new class of RL algorithms, based on opposition. By considering states and opposite states as well as actions and opposite actions simultaneously, multiple updates can be made. This leads to a shorter exploration period. Therefore, a desirable level of accuracy for the optimal action policy can be achieved in a shorter time.

In order to explain how the concept of opposition-based learning can be used to extend reinforcement agents, we focus on the simplest and most popular reinforcement algorithm, Q-learning. In this algorithm, the amount of time needed for convergence is proportional to the size of the Q-matrix. A larger Q-matrix, resulting from a larger number of states and/or actions, requires more time to be filled. Generally, the RL agent begins with no knowledge and makes stochastic decisions, explores the environment, finds rewarding actions and exploits them. In particular, at the very beginning, the performance of the RL agents is poor due to a lack of knowledge about which actions can steer the environment in the desired direction. Whenever the RL agent takes an action it should also consider the opposite action and/or opposite state. This will shorten the state-space traversal and should consequently accelerate the convergence. The concept of opposition can of course be applied if opposite actions and opposite states are known or meaningful in the context of the problem at hand. In regard to action a and state s and the existence of their opposites $\breve{a}$ and $\breve{s}$, the following cases can be distinguished [85]:

1. Opposite action $\breve{a}$ and opposite state $\breve{s}$ are given: at least four cases can be updated per state observation.

2. Only $\breve{a}$ can be defined: two cases can be updated per state observation.

3. Only $\breve{s}$ can be defined: two cases can be updated per state observation.

4. Neither $\breve{a}$ nor $\breve{s}$ can be given: application of opposition concept not straightforward.

Assuming that opposite action $\breve{a}$ and opposite state $\breve{s}$ both exist, then at least four state-action pairs can be updated in each iteration. In general, if action a is rewarded for state

s, then a is punished for the opposite state $\breve{s}$, the opposite action $\breve{a}$ is punished for s and rewarded for $\breve{s}$ (Figure 3.3).

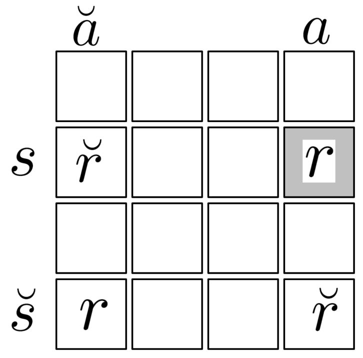

Figure 3.3: Time saving in RL: the action a is rewarded for the state s. The opposite cases are updated simultaneously without explicit action-taking.

In order to make additional updates as described, the RL agent has to know how to find opposite actions, and opposite states. Clearly, this will depend on the application at hand. Whereas for some applications opposite actions are straightforward, this may not be the case for others. Nonetheless, general procedures may be defined to facilitate this. A degree of opposition $\breve{\varrho}$ can be defined to measure how far two actions a_1 and a_2 are the opposite of each other [85]:

$$\breve{\varrho}(a_1|_{s_i}, a_2|_{s_j}) = \eta \times \left[1 - e^{\left(- \frac{|Q(s_i,a_1) - Q(s_j,a_2)|}{\max\limits_k \left(Q(s_i,a_k), Q(s_j,a_k) \right)} \right)} \right], \tag{3.11}$$

where η is the state similarity and can be calculated based on state clustering [85].

Considering action a_t, when visiting state s_t, opposition-based Q-learning for opposite action can be defined as given in Figure 3.4.

```
Initialize $Q(s,a)$ randomly
Repeat (for each episode)
    Initialize $s$
    Repeat (for each iteration of episode)
        Choose $a$ from $s$ using policy $\pi$ derived from $Q$
        Take action $a$, observe reward $r$, and next state $s'$
        $Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma \max_{a'} Q(s',a') - Q(s,a)]$
        Calculate the degree of opposition(optional)
        Determine opposite action $\breve{a}$ and next state $s''$
        Calculate the opposite reward $\breve{r} = p$
        Determine next opposite action $\breve{a'}$ from $s''$
        $Q(s,\breve{a}) \leftarrow Q(s,\breve{a}) + \alpha[\breve{r} + \gamma \max_{a'} Q(s'',a') - Q(s,\breve{a})] \leftarrow Q(s,\breve{a}) + W.\breve{r}$
        $s \leftarrow s'$
    until $s$ is terminal
```

Figure 3.4: Opposition-based Q-Learning for opposite action [84].

3.5 Chapter Summary

Reinforcement learning is based on the idea that an intelligent agent learns through trial-and-error interactions with its environment. It can continuously learn and eventually decide to take the actions that are most rewarding. RL has already been used in some areas such as robotics and game-playing. When RL is applied to image-based tasks, a large number of state-action pairs is involved. The OBL idea is used to extend RL agents to shorten the exploration time.

In the next chapter, the proposed Reinforced Image Segmentation (RIS) system will be introduced.

Chapter 4

Reinforced Image Segmentation

4.1 Overview

Many image-processing applications need to extract just *one object of interest*. Segmentation techniques are used for this purpose, but many of these techniques extract the objects based on a static scheme without any intelligent and ongoing change in the segmentation parameters. In order to establish an intelligent framework, it is first needed to localize the significant parameters in a segmentation processing chain. These parameters should be subsequently adjusted when the image and object conditions vary.

In this chapter, an adaptive segmentation system based on opposition-based reinforcement learning is introduced that can learn and adjust itself. By using an RL agent in two modes, this method can find the appropriate local values for image processing tasks to segment an object. The goal of this agent-based system is to identify one object of interest in an image and separate it from the background and other objects. The proposed reinforced image segmentation (RIS) is a general framework providing a systematic approach to design an intelligent segmentation system. To speed up the learning procedure, opposition-based theory is employed to reduce the computational cost.

4.2 Overall RIS Structure

The adaptive parameter control of a multi-task segmentation system is a challenging problem. The efficiency and robustness of a learning technique are two important issues, if applied to such a system. The algorithm must be able to acquire effectively domain knowledge and adapt itself to new conditions for each input image. In the proposed RIS system, a general framework for a multi-stage image segmentation system based on reinforcement learning is constructed. Due to the nature of tabular reinforcement learning, in terms of state, action and reward definitions and their interactions with each other, the proposed system is able to adapt to the size, location and rotation of the object, as well as to variation in the background characteristics. A multi-stage structure, based on RL, allows us to recognize the appropriate parameters for the entire processing. This system is introduced as a general framework. To speed up the learning process, we use opposition-based learning to perform more updates in each iteration.

Figure 4.1 shows the general structure and components used in the proposed system. First, a digital image is acquired as input. Then, the image quality is improved through a preprocessing step. The output of the preprocessing step goes to the main section, which is a collection of segmentation tasks. This step contains several tasks (each applied on a sub-image), with some parameters to be adjusted. The selection of these tasks depends on the application at hand. The output of this step is represented as a labelled image containing some objects. The role of the last step, object extraction, is to find the object that is meaningful for our purpose. The section with the RL agent is responsible for learning and adjustments. Interaction with the user is included in this section to integrate

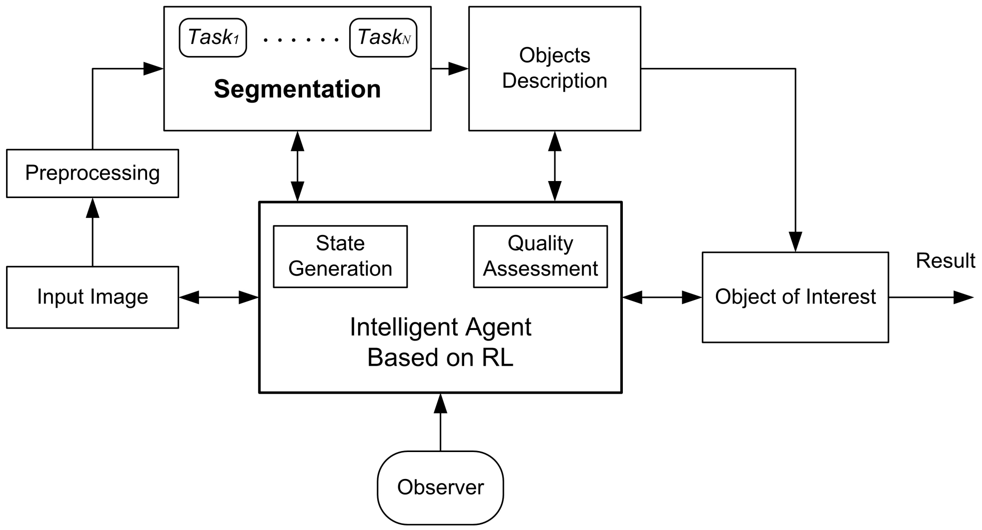

Figure 4.1: Structure and components used in RL-based segmentation system.

prior information, as well as the subjective evaluation. To understand the link between reinforcement learning and image segmentation, it should be noted again that the RL module receives a reward or punishment from its environment, depending on the action taken. The RL agents discover which actions bring more reward using exploration and exploitation. In a deterministic environment the agent tries to find the best action, while in a nondeterministic environment it tries to track the optimal action, one which may change over time.

Figure 4.2 illustrates the overall structure of the learning modes in the RIS system. In the offline mode, the RIS system is trained using manually segmented images represented as “ground-truth images”. The RL agent can acquire the necessary knowledge using these sample images and fill the Q-matrix. Using this knowledge, the agent can recognize those actions with maximum rewards for each state. In this mode, the user feedback can be used as an initialization procedure to provide some basic information for the system. The

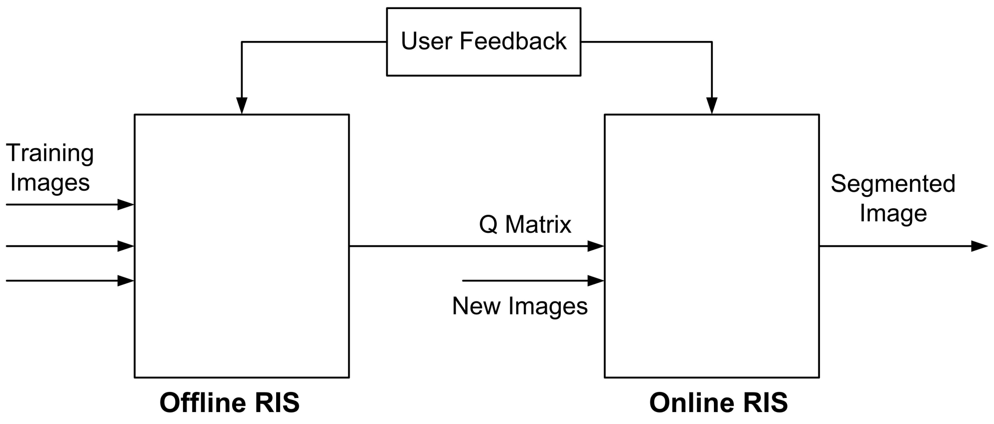

Figure 4.2: Overall RIS system and two learning modes.

system then switches to the online mode, where new input images are processed. The agent can choose the appropriate actions for each image based on its accumulated knowledge. Furthermore, it can continue to learn and increase its performance if the accumulated knowledge does fail to segment the image properly. The structure used in this system can also incorporate subjective feedback as a corrective signal. If no ground-truth exists, the RIS can proceed to the next stage. However, the subsequent learning time will dramatically increase.

4.2.1 Parameter Adjustment

The goal of the RIS system is to adjust the parameters of the processing tasks, such that the evaluation metric for the final output becomes maximal. The processing tasks in the chain are chosen based on the desired output. If we want to employ an intelligent architecture to find appropriate parameters for these tasks, we may encounter the following obstacles:

- The large number of possible cases for parameter adjustment may lead to assigning incorrect values to some of them.

- Relying on objective evaluation as the only assessment may not be reliable for an image-based problem.

Through interaction with the environment, reinforcement learning has the ability to overcome the above obstacles:

- Through an exploratory policy, the agent can be prevented from going to the local maxima.
- An RL agent can easily employ subjective evaluation in terms of user feedback.

The processing tasks involve low-level features to segment the image. These features are pixel-based information that can be extracted locally from sub-images, or globally from the entire image. The object description, however, involves extraction and analysis of global shape characteristics generally based on prior information. There is a variety of knowledge that can be used to aid different steps. By incorporating user interaction, as a part of the RL agent section, the overall performance of the system can be improved. This, however, should be implemented as an indirect and optional feedback to reduce the burden on the user.

There is an original input image I to the system. There are also N processing tasks $\{Task_1, Task_2, ..., Task_N\}$ with the output images $I_{O_1}, I_{O_1}, ..., I_{O_N}$ to segment the input image. Before applying the processing tasks, we may have one or more pre-processing steps. Therefore, the input image to the processing task section is indicated by $\hat{I}$ that is the original image after pre-processing (e.g., noise filtering, contrast enhancement, etc.). Each task has some control parameters as well as some features in its input and output images. For each processing task, the RL agent can observe the situation of different steps using a set of features $\mathcal{X} = \{x_1, x_2, ..., x_T\}$ extracted from the input and output images, and change the control parameters $\mathcal{C} = \{\mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_N\}$ to achieve the desired results. To identify the best parameter set, the system adaptively adjusts the control parameters to optimize its performance for a given segmentation problem. We will see that an RL agent can find these parameters by using the experience accumulated during the progressive learning process and taking appropriate actions, such that the proposed RIS system has the following characteristics:

- Requiring a limited number of training data.
- Requiring a practical level of user interaction.
- Improving performance permanently.

4.3 RIS Design Aspects

As stated, the goal of RIS is to construct a segmentation system that adaptively adjusts the parameters of a multi-task algorithm using an RL agent. This system incorporates the knowledge from input, output and intermediate images.

In the proposed RIS system, the input image I is divided into L_s sub-images with the size $\omega_R \times \omega_C$. Due to the changes in the characteristics of sub-images, each sub-image is a non-deterministic environment for the RL agent. That is because after sensing a state s and taking action a, the reward may be r' and the next state s'; while on another occasion, employing another sub-image, after sensing state s and taking action a there may exist a different reward r'' and a different next state s''. Limiting ourselves to the model described in the previous chapter, the state s, action a and reward r are defined first. To define the state, some features of the input ,as well as output images after each processing task are used. The results of the feature extraction are used to generate the states for the RL agent.

To define the actions, the idea of the actions being modifications of the parameters used in processing tasks is considered. The agent adjusts these parameters and receives a reward. The reward is an external signal provided by the environment to guide the learning process. When the agent has modified the parameters of the processing tasks, we can evaluate the output image to produce a reward.

4.3.1 Defining the RIS States

We map the features $x_1, x_2, ..., x_T$ which represent the status of the image in each step of processing task to the state space $\mathcal{S} = [\xi_1 \quad \xi_2 \quad ... \quad \xi_q]$ where ξ_i are q values obtained from the features x_i. For example, if there are two features x_1, x_2 so that each of them can take the values $\{1, 2, 3\}$ the possible states can be obtained as indicated in Table 4.1. The

Table 4.1: Two sample features and their corresponding states.

State	**Feature** x_1	**Feature** x_2
1	*1*	*1*
2	*1*	*2*
3	*1*	*3*
4	*2*	*1*
5	*2*	*2*
6	*2*	*3*
7	*3*	*1*
8	*3*	*2*
9	*3*	*3*

potential features that could be used in this case are:

- Features quantifying some geometrical measures for each sub-image such as relative position with respect to particular landmarks,
- Features which reflect the local information in gray level input images such as first, second or higher order of spatial distribution. Local histogram or texture information can be included in this group, and
- Features which reflect the object properties after applying the processing tasks. Various shape or boundary properties such as area, Euler number, compactness, convexity, slope density function, curvature, and edge properties, assuming a binary image after the processing task, are in this group.

If $s_{\ell,i}$ represents the state for sub-image number $\ell = \{1, ..., L_s\}$ in iteration i, we can define it as the following function:

$$s_{\ell,i} = f(P_\ell, G_\ell, B_{\ell,i}) \tag{4.1}$$

where P_ℓ, G_ℓ and $B_{\ell,i}$ are three feature value sets showing the sub-image geometrical characteristics, gray level information and properties of binary output, respectively.

Due to applying the method on each sub-image, employing those features which result in an extremely large state space is not efficient. Suitable feature selection is highly dependent on the specifications of the problem at hand, as this is the case for virtually all classification or clustering-based methods. In Chapter 5, state definition will be concretized for specific applications.

4.3.2 Defining the RIS Actions

In order to extract the object of interest, the action set $\mathcal{A}$ should contain actions that properly change each control parameter set $\mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_N$ for each individual sub-image. The system can change the assigned values, or choose among predefined values in a specific set to control the effect of each processing task. Hence, we have a set of actions $\mathcal{A}$ where each them stands for operations to change parameter sets $\mathbf{c}_1, ..., \mathbf{c}_N$ for tasks $1, ..., N$.

When the system is in a specific state $s_{\ell,i}$ (for sub-image ℓ in iteration i), taking an action $a_i \in \mathcal{A}$ brings the system to the next state $s_{\ell,i+1}$. If $c_{\ell,i}$ represents the control parameter set for sub-image number ℓ in iteration i, it is also changed to the new set $c_{\ell,i+1}$ after the action a_i is taken. For the applications discussed in Chapter 5, the actions will be defined explicitly.

4.3.3 Defining the RIS Reward

By taking the action $a_i \in \mathcal{A}$ for sub-image ℓ, the agent receives a reward $r_{s_i,s_{i+1}}$ depending on the results of the transition from state $s_{\ell,i}$ to $s_{\ell,i+1}$. The reward is defined according to the quality criterion that represents how well the object is segmented in each sub-image. For the training in the offline mode, a straightforward method is comparing the result of the system with the ground-truth image (manually segmented image) after each action. To measure this value for each sub-image, we note how much the quality has changed after the action. In each sub-image, to improve the quality of the segmented object, the agent receives a reward; otherwise it is punished. The dissimilarity, D_i, between the result of the system for sub-image I_{i_s} and its desired (ground-truth) output $I_{i_{sd}}$ is defined as

$$D_i = \frac{XOR(I_{i_s}, I_{i_{sd}})}{\omega_R \times \omega_C}, \tag{4.2}$$

where XOR indicates the number of mismatched pixels between these two sub-images, and ω_R and ω_C are the number of rows and columns of the sub-image. Therefore, the general form for the reward function is represented by

$$reward = \begin{cases} R_1 & D_i^{AFTER} < D_i^{BEFORE}, \\ R_2 & otherwise, \end{cases} \tag{4.3}$$

where D_i^{AFTER} and D_i^{BEFORE} indicate the dissimilarities with respect to the ground-truth image before and after taking the action, respectively. R_1 and R_2 are constant values (i.e. 1 and 0, or 1 and -1).

Although the online reward is mainly subjective through user feedback, an objective evaluation is also performed to provide an automatic assessment before receiving the subjective feedback. This will be explained in detail in Section 4.5.

Figure 4.3 illustrates the relationship between states, actions and reward, based on the above notation in a transition diagram for a sample sub-image ℓ from initial to final state.

4.4 Offline RIS

The offline training mode is conducted using the manually segmented samples. The general scheme for the offline RIS system is shown in Figure 4.4. In this mode, the RL agent interacts with $M \geq 1$ known cases to acquire fundamental information about the image characteristics and involved segmentation parameters. Two different situations can be distinguished regarding the offline mode:

- The RL agent may proceed to online mode, while using the experiences it has obtained in the offline mode, continuing to improve its behaviour through interaction with the online environment;

- The knowledge acquired during the offline mode can be applied on the real-time cases without any further interactions with the environment. In this situation no

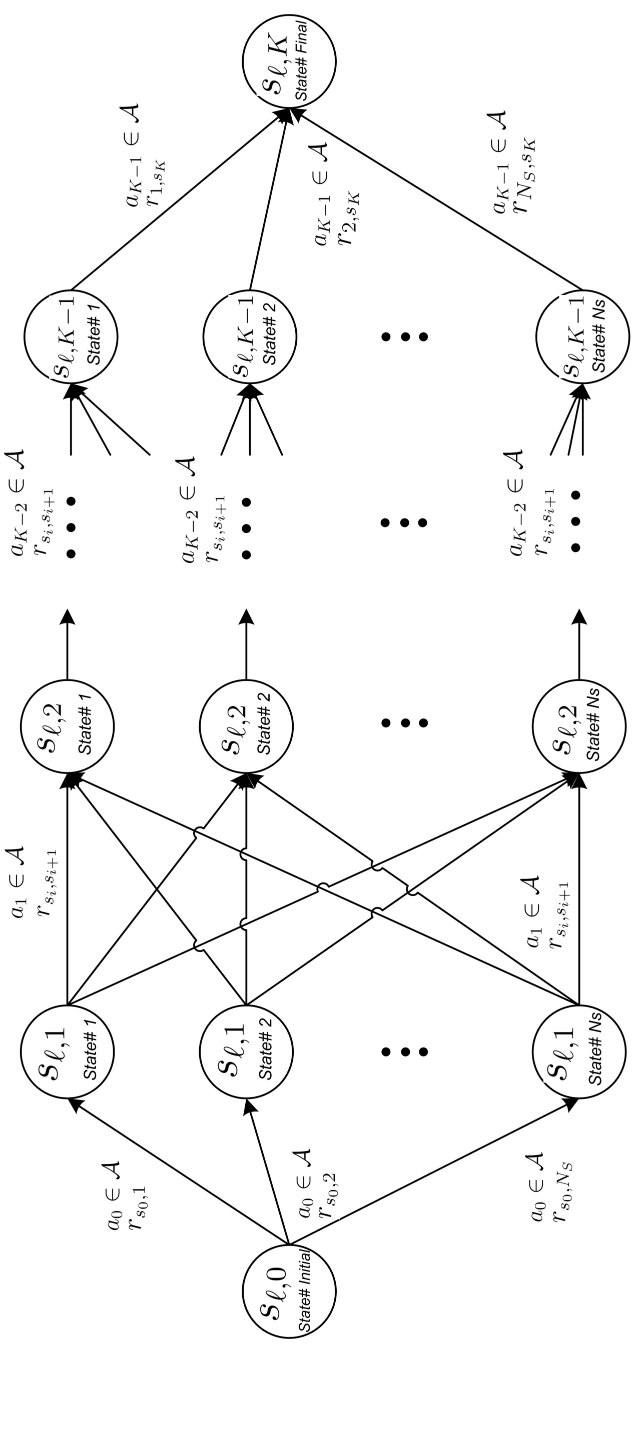
$s_{\ell,0}$ State# Initial
$a_0 \in \mathcal{A}$ $r_{s_0,1}$
$a_0 \in \mathcal{A}$ $r_{s_0,2}$
$a_0 \in \mathcal{A}$ r_{s_0,N_S}
$s_{\ell,1}$ State# 1
$s_{\ell,1}$ State# 2
$s_{\ell,1}$ State# Ns
$a_1 \in \mathcal{A}$ $r_{s_i,s_{i+1}}$
$a_1 \in \mathcal{A}$ $r_{s_i,s_{i+1}}$
$s_{\ell,2}$ State# 1
$s_{\ell,2}$ State# 2
$s_{\ell,2}$ State# Ns
$a_{K-2} \in \mathcal{A}$ $r_{s_i,s_{i+1}}$
$a_{K-2} \in \mathcal{A}$ $r_{s_i,s_{i+1}}$
$a_{K-2} \in \mathcal{A}$ $r_{s_i,s_{i+1}}$
$s_{\ell,K-1}$ State# 1
$s_{\ell,K-1}$ State# 2
$s_{\ell,K-1}$ State# Ns
$a_{K-1} \in \mathcal{A}$ r_{1,s_K}
$a_{K-1} \in \mathcal{A}$ r_{2,s_K}
$a_{K-1} \in \mathcal{A}$ r_{N_S,s_K}
$s_{\ell,K}$ State# Final

Figure 4.3: General state transition diagram for the sub-image ℓ from initial state (s_0) to final state (s_K) .

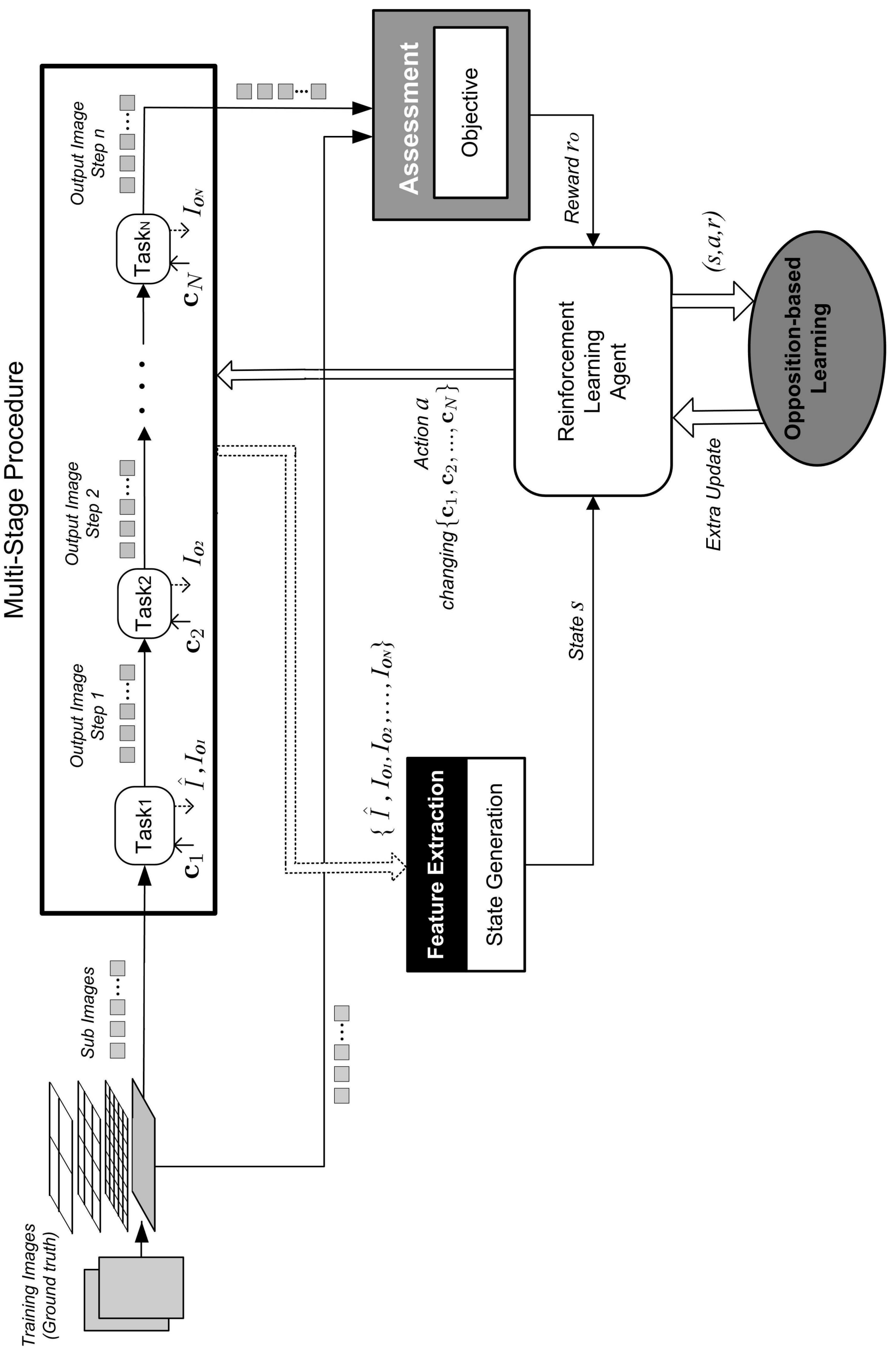

Figure 4.4: The general scheme for offline RIS system.

additional learning will be carried out and the accumulated knowledge, stored in a lookup table, will be used to segment images.

Considering the nature of an RL agent, if we pursue the first option, we can perform the learning stage using a limited number of training samples and gain extra knowledge during the online mode. The second option would be appropriate only if the dynamics of the application does not vary.

As stated, the input image I is divided into L_s sub-images. We follow a hierarchical procedure from low to high-resolution for the size of sub-images. The system starts with large sizes of sub-images (low-resolution) and proceeds until the best match between the extracted object and the ground-truth is achieved. Due to the low resolution, the obtained control parameter values are most likely not satisfying in some parts, but at least they can be used as initial values for the next resolution with a smaller sub-image size. Again, the image is divided into sub-images, but in smaller size (higher resolution), and with the initial parameter values gained from the previous resolution. This procedure continues until the desired sub-image size is achieved. Figure 4.5 shows a schematic view of this hierarchical procedure. By using this method, the agent achieves some sub-goals before going to the finest resolution for its final goal. As a result, it can increase the overall speed. The RL agent works on each sub-image separately to find the best parameter match.

During the RIS offline training, where we have the desired output image, the agent works on each sub-image and explores the solution space. In this mode, the agent tries different actions in different states using an exploratory procedure. The RL agent changes the parameters for each sub-image. By taking each action the agent receives a corresponding reward for that state-action pair and updates the corresponding value in the Q-matrix. After this mode, the agent has explored many actions. The procedure implemented in the offline mode is described in Figure 4.6. This figure shows the relation between states and

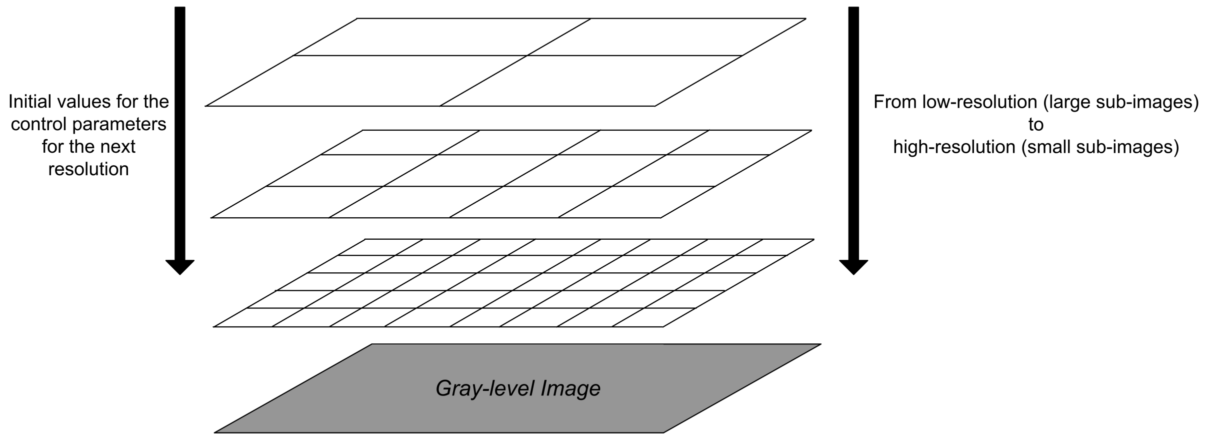
Initial values for the control parameters for the next resolution
From low-resolution (large sub-images) to high-resolution (small sub-images)
Gray-level Image

Figure 4.5: Hierarchical model used to speed up the offline learning mode. The initial values for the control parameters are provided from upper level (large sub-images) to lower level (small sub-images).

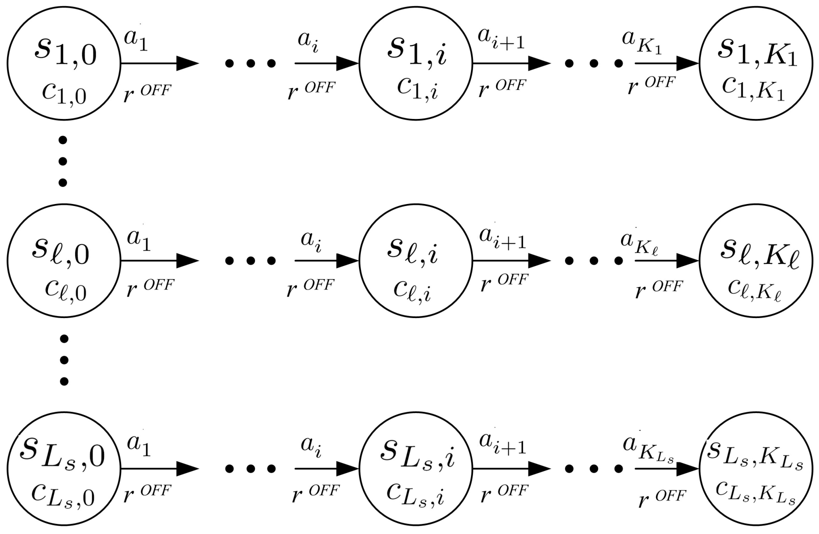

Figure 4.6: State transition diagram for different sub-images in the offline mode.

actions in a transition diagram. In this mode, the ground-truth images are available. The RL agent works on each sub-image ℓ individually. It continues until a criterion is met or a maximum number of iterations K is reached. After taking one action a_{i+1} in state $s_{\ell,i}$ the system goes to the next state $s_{\ell,i+1}$ and the agent receives an objective reward r^{OFF} based on the segmentation quality for that sub-image. The agent then updates its knowledge. This process is performed for all sub-images $\ell = \{1, ..., L_s\}$ individually, as shown in the state transition diagram. For quality measure of each sub-image we calculate the similarity with the ground-truth. To measure this similarity we can calculate the overlap between the ideal output and the binary image generated by the RL agent.

During this procedure, the agent must explore the solution space. This can be achieved using the Boltzman policy with a high temperature or ϵ-greedy policy [79].

Figure 4.7 describes the summary of the algorithm used in the offline mode.

As stated, when we integrate the RL agent into image-based applications, an obstacle may be the large number of state-action pairs involved in the processing tasks. In these applications, usually high computational expense is required to solve the problem. We use the opposition-based learning theory to overcome this problem. Depending on how the state-action pairs are defined, this technique can be used for states and/or actions. During this procedure, the agent explores the parameter space more rapidly.

4.5 Online RIS

Once the agent has learned all available M training samples, it proceeds to the online mode. In this mode, the RL agent is used in real-time to segment new images. Figure 4.8 illustrates the general scheme for the online RIS system. The agent learns continuously in this mode and can permanently adapt to the new conditions of input images. A subjective

Offline mode for training images

```
Select a training image
  Create the sub-images
  Set for more exploration policy (e.g. using $\epsilon$)
  Repeat for all sub-images
      Extract the sub-image $\ell$
      Initialize $s$ using initialized parameters $\mathbf{c}_i$
      Repeat
          Calculate the state $s$ using features $[x_1 \quad x_2 \quad ... \quad x_T]$
          Take the action $a_i$ for processing tasks $1, ..., N$ based on the
          policy $\pi$ derived from $Q$
          Observe the objective reward $r^{OFF}$ using ground-truth and
          calculate the next state $s'$
          Update the Q-matrix
      Until a maximum number of iterations $K$ is reached or a
      specific criterion is met
  Until all $L_s$ sub-images are visited
Until all $M$ training images are learned
```

Figure 4.7: Summary of the algorithm used for offline RIS.

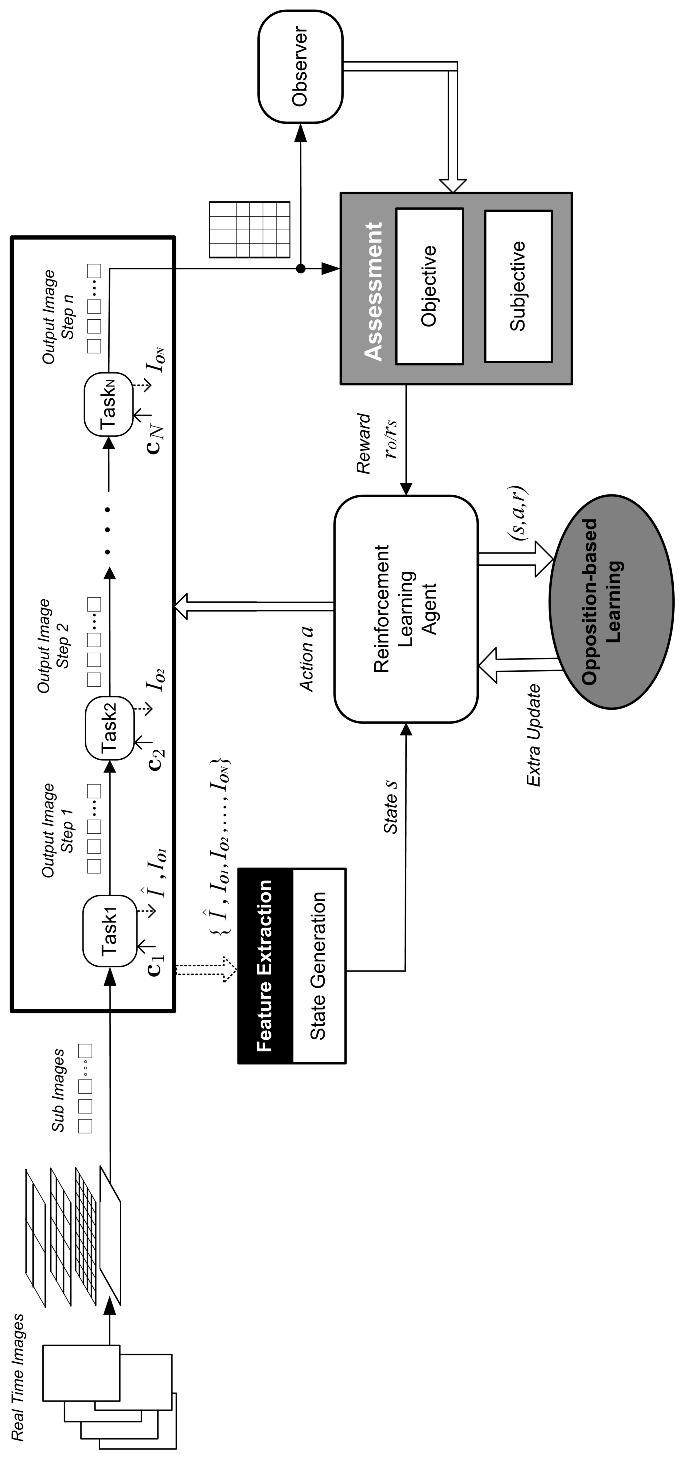

Figure 4.8: The general scheme for online RIS system.

user feedback can also be provided in this mode.

In the online mode, we have a global perspective on the system. This relies on a model-matching procedure to provide evaluation and improve the results of the adaptive segmentation system. Global information must be incorporated to evaluate the capability of performing segmentation which uses local control parameters. This is performed using two strategies: top-down and bottom-up. The bottom-up strategy uses local features to find the regions belonging either to the object or to the background, while the top-down uses object representation to detect it in the image. It means that the global characteristics of the object of interest are used to modify the local parameters of segmentation. For example, when some pixel groups, showing different types of statistical homogeneity, belong to the same object, a top-down view gives required information to the system so that they can be merged.

In the RIS system, after finding the global characteristics, the system switches to local processing to update its knowledge and optimize segmentation performance where it simultaneously considers the global features. From the training images used in offline mode, a priori knowledge of the approximate shape of the object of interest is available. This knowledge is also updated using the images that have passed the subjective confirmation in the online mode. After performing the processing tasks and creating the binary output image, the object with the highest match is taken as the candidate to the learning system. A global model matching method must be used to detect the difference between the extracted object and the approximate shape of the object of interest (available form training images). By detecting these differences, the system can find the sub-images that have inappropriate control parameters. Subsequently, the system provides an objective punishment for the action resulting in those parameters.

After the objective evaluation, the subjective evaluation can be performed where the user assesses the results of the system. For each sub-image, the local adjustment procedure terminates when no more changes are performed by the user. This leads the RL agent to an episodic procedure, where during each iteration the agent works on each sub-image and takes the action to change its state. Then it proceeds to the next sub-image and performs the same procedure until it completes the entire image for that iteration. For each sub-image, the reward is not calculated immediately. Instead, the agent waits until the whole image is scanned (in one iteration) and global evaluation (objective and/or subjective) is performed. It is then provided with a reward for each sub-image. This is calculated based on a quality criterion representing how well the object has been segmented in each sub-image before and after the action. This is specifically useful where images have widely the same object and background characteristics so that for each input image, the learning process uses the knowledge obtained from previous input images.

Figure 4.9 shows the state transition diagram for the online RIS.

In this mode, in each iteration i the RL agent takes the appropriate action and finds the assigned control parameters $c_{\ell,i}$ for each sub-image ℓ, proceeding then to the next sub-image ℓ (from the top to the bottom of diagram 4.9) until it scans the entire image in one episode. The main difficulty in a local process is that an object may be split into multiple regions (over-segmentation), or some irrelevant regions may merge the object with its background (under-segmentation). The global perspective provides useful information to overcome this problem and helps achieve an accurate delineation of object boundaries. In each iteration, we segment all sub-images into regions using the knowledge that the system has already acquired and complete an episode, but after each episode, we consider the characteristics of bounding contours in a global manner to evaluate the results. The RL agent can then use the results of this evaluation as an online rewards r^{ON} for each action taken on each individual sub-image. If a sub-image is segmented correctly in terms

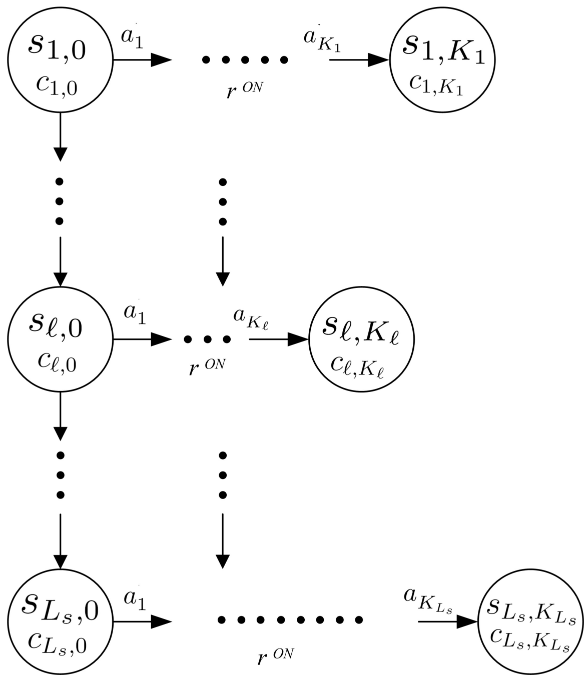

Figure 4.9: State transition diagram for different sub-images in the online mode.

of global feedback (objective or subjective), the parameters are left unchanged for that sub-image and the process is terminated. The online reward is a crucial assessment for the RIS system. The summary of the algorithm used in the online mode is described in Figure 4.10.

```
Online mode for test images
Select a test image
  Set for more exploitation (e.g. using $\epsilon$)
  Repeat
      Initialize $s$ using initialized parameters $\mathbf{c}_i$ for all sub-images
      Repeat for each sub-image
          Extract the sub-image $\ell$
          Calculate the state $s$ using features $[x_1 \quad x_2 \quad ... \quad x_T]$
          Take the action $a_i$ for processing tasks $1, ..., N$ based on the
          policy $\pi$
      Until all $L_s$ sub-images are visited
      Evaluate the result objectively
      Calculate the online reward $r^{ON}$ objectively
      Evaluate the result Subjectively
      Calculate the online reward $r^{ON}$ subjectively
      Update the Q-matrix
  Until the user confirms the result (No manual change is made)
Until all test images are segmented
```

Figure 4.10: Summary of the algorithm used for online RIS.

Objective Evaluation

We want to apply an objective evaluation as primary assessment. One way is to use algorithms that compute a measure of distance between two shapes. These algorithms usually compute the global best match between two shapes by pairing up each element of shape Γ_1, with an element of shape Γ_2 to give the minimum difference. The transformations used in these algorithms are generally invariant to translation, scaling, and rotation, preserving the information contained in the curve. We have used the algorithm introduced in [58] in a 2-D plane to measure the similarity. We suppose that there is an unknown curve

$$C^1 = \{p_1^1(x_1^1, y_1^1), p_2^1(x_2^1, y_2^1), ..., p_n^1(x_n^1, y_n^1)\},$$

and a prototype curve

$$C^2 = \{p_1^2(x_1^2, y_1^2), p_2^2(x_2^2, y_2^2), ..., p_m^2(x_m^2, y_m^2)\}$$

corresponding to the shape Γ_1 and Γ_2, respectively, and $p_{i_p}^{\{1,2\}}(x_{i_p}^{\{1,2\}}, y_{i_p}^{\{1,2\}})$ are the points located on these curves. If we represent the characteristics of the points located on two curves as strings $\tilde{x_1}$ and $\tilde{x_2}$, the distance between two strings is defined as the shortest way (minimum cost) that transforms $\tilde{x_1}$ into $\tilde{x_2}$. The main idea behind the algorithm is based on finding a minimum cost path as a sequence of operations, transforming a string which corresponds to the transition of C^1 into C^2. To make this transformation, the algorithm allows changing one element of a string into another element, deleting one element from the string, or inserting an element into the string. Using these operations, a measure of similarity between two strings is computed. The changes performed in this measurement correspond to the insertion, substitution, and deletion operations. The algorithm not only gives the distance $\Delta(\tilde{x_1}, \tilde{x_2})$ between two strings $\tilde{x_1}$ and $\tilde{x_2}$, but also the way in which the symbols of $\tilde{x_1}$ correspond to those of $\tilde{x_2}$. We may use the signature of a shape to generate its string. Generally speaking, a shape signature is any 1D function representing 2D areas or boundaries. Complex coordinates, central distance, curvature and cumulative angles are among the commonly used signatures in literature, and are shown to be useful for shape representation [95].

In our approach we can calculate the central distance as the distance from the points on the boundary to the centre of the object, represented as a 2π periodic function. This method is especially useful when we can calculate a reliable central point of the object (e.g., represented by the user). In this case, one angle θ_C is assigned to a distance d (nearest corresponding contour point) represented as a function $d = f(\theta)$ to generate the central distance signature. It should be noted that if the curvature is used as a signature, then the boundary of the object must be sufficiently smooth. That is because the curvature is very sensitive to noise.

The proposed method compares the signature of the extracted object to the standard signature of the object we are looking for. We can extract the standard signature using the ground-truth images from the offline mode. Using this method, we can estimate significant deviations and provide the objective reward (/punishment) r^{ON} for the corresponding sub-image.

Subjective Evaluation

Another alternative is subjective evaluation through online feedback from the user. The user looks at the result of the system for each image and if he/she is not satisfied, he/she may change the result manually by adjusting some points of the extracted object boundary. These changes are evaluated as subjective punishment for the agent, which can take the manually corrected result and use it as a new ground-truth image to improve its knowledge.

From a global viewpoint, this is a top-down process. Hence, the agent can be trained further online through a subjective evaluation r^{ON}_{SUBJ}. The system adapts itself via updating Q-matrix and follows the changes in the new input images if unvisited states are encountered. If the user saves the RIS result without any change, the system result can then be adopted as a new ground-truth to update the information for the RL agent.

4.6 Chapter Summary

In this chapter the proposed RIS system, using an opposition-based reinforcement learning scheme, was presented. RIS finds the appropriate local values for image processing tasks and segments an object. Opposition-based theory is employed to reduce the training time. During an offline stage, the agent uses some images and their manually segmented versions to gather appropriate knowledge. The agent is provided with scalar reinforcement signals to explore/exploit the solution space. The agent can then use the acquired knowledge to segment similar images. Also, it can improve itself using online feedback.

In the next chapter, we customize the general configuration of the proposed RIS system for some image segmentation applications and provide experimental results.

Chapter 5

Experimental Verifications

In the previous chapter, the generic RIS system was established. The goal of this agent-based system is to identify one object of interest in an image and separate it from the background and other objects by offline training and continuous online learning. In this chapter three case studies will be described where such a system can be applied. The general framework proposed in Chapter 4 will be implemented in three different configurations, in order to demonstrate the potentials of the RIS system.

5.1 RIS for Global Object Extraction

As a special case for the proposed system we consider a global segmentation task, which should extract a specific object. To formulate this problem within the RIS context, we define only one sub-image with the same size as the original image. This is a special case, when the RIS system is implemented globally to operate on the entire image. We assume that a global threshold value exists which can extract our object, or at least a large, recognizable portion of it. Other irrelevant objects may also be revealed by setting that threshold, but these can be removed via further processing. The global RIS adjusts the threshold value and the size of morphological closing as a post-processing operator. Other thresholding techniques, such as Kittler [35] and Otsu [47] methods, will be implemented as well, in order to assess comparatively the results of the global RIS.

5.1.1 RIS Configuration

The first step in extracting the object of interest is to threshold the image. We assume that there is a non-uniform illumination across the image. Due to possible imperfections, there generally exist over- or under-segmentation. A morphological closing can be used to post-process the thresholded image and create a connected and complete object. The structure of the RIS system for this problem is based on the framework introduced in Chapter 4 and processing tasks are shown in Figure 5.1. The characteristics of the system can be represented as follows:

- In terms of algorithms there are two processing tasks, global thresholding and morphological closing. The parameters of these tasks are adjusted by the RIS system.
- The processing methods operate globally on the entire image.
- For learning, the knowledge acquired during the offline mode is stored and applied on the real-time cases with no further online learning (lookup-table approach).

First, three RIS components should be defined: states, actions and reward.

States

The following features are selected to define the state:

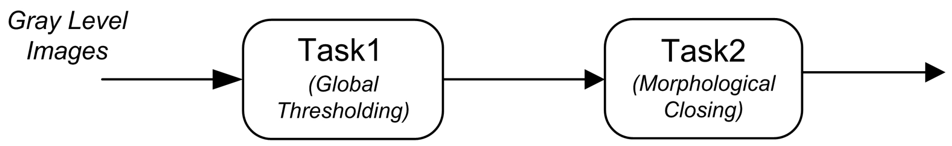

Figure 5.1: Processing tasks for RIS system for global object extraction. The framework is based on the system introduced in Chapter 4(see Figure 4.4, page 54).

Feature 1: *Elliptic Fourier Descriptors*

A shape descriptor, representing the pixel arrangement on the boundaries of the revealed objects, is the first feature. This descriptor is based on Fourier characterization of a boundary, with a set of coefficients representing its frequency content. The descriptor is invariant with respect to the rotation, relocation and changing of the boundary starting point. To find the Fourier descriptor, a suitable curve representation must be defined. We use elliptic Fourier descriptors, which represent a curve in a two-dimensional space. This can be achieved if we consider the image space as a complex plane. In this plane, we can define the boundary of an object as [19]:

$$c(u) = x(u) + jy(u). \tag{5.1}$$

The variable u is obtained by the arc-length parameterization and changed from 0 to 2π to cover a whole period. The Fourier expansion of $c(u)$ is defined using a set of infinite complex exponential components as [19]:

$$c(u) = \sum_{k=-\infty}^{\infty} c_k e^{jk\omega u}. \tag{5.2}$$

In this general form c_k is the elliptic coefficient and ω is the fundamental frequency. The coefficients can be represented with real and imaginary parts:

$$c_{xk} = \frac{a_{xk} - jb_{xk}}{2}, \quad c_{yk} = \frac{a_{yk} - jb_{yk}}{2}, \tag{5.3}$$

where

$$a_{xk} = \frac{2}{m_b}\sum_{i=1}^{m_b} x_{u_i}\cos(ki\omega\upsilon), \quad a_{yk} = \frac{2}{m_b}\sum_{i=1}^{m_b} y_{u_i}\cos(ki\omega\upsilon), \tag{5.4}$$

and

$$b_{xk} = \frac{2}{m_b}\sum_{i=1}^{m_b} x_{u_i}\sin(ki\omega\upsilon), \quad b_{yk} = \frac{2}{m_b}\sum_{i=1}^{m_b} y_{u_i}\sin(ki\omega\upsilon). \tag{5.5}$$

x_{u_i} and y_{u_i} are discrete values of the function $x(u)$ and $y(u)$, respectively. m_b is the number of discrete points on the object contour and $\upsilon = 2\pi/m_b$. The following equation is now used to create a unique vector based on the four fundamental components a_{xk}, a_{yk}, b_{xk} and b_{yk} [19]:

$$CE(k) = \left(\frac{a_{xk}^2 + a_{yk}^2}{a_{x1}^2 + a_{y1}^2}\right)^{\frac{1}{2}} + \left(\frac{b_{xk}^2 + b_{yk}^2}{b_{x1}^2 + b_{y1}^2}\right)^{\frac{1}{2}}. \tag{5.6}$$

The N_F first and most significant $CE(k)$s are chosen and stored in a vector CE_N. This vector is used in the following stages as the representation of the object contour. CE_N for the reference object is calculated as CE_{ref}. In each iteration, the CE_N and the normalized difference with CE_{ref} are determined for each revealed object to calculate the first feature:

$$CE_\Delta = \frac{\left(\sum\limits_{n=1}^{N}(CE_{ref}(n) - CE_N(n))^2\right)^{\frac{1}{2}}}{\left(\sum\limits_{n=1}^{N}(CE_N(n))^2\right)^{\frac{1}{2}} + \left(\sum\limits_{n=1}^{N}(CE_{ref}(n))^2\right)^{\frac{1}{2}}}, \tag{5.7}$$

where $CE_\Delta \in [0,1]$. The smaller the CE_Δ, the higher the similarity of the extracted object to the reference object.

Feature 2: *Area*

The second feature used to define the state, is the area of the extracted object. We calculate A_{ratio} as the area ratio of the extracted object with respect to the reference object:

$$A_{ratio} = \frac{A_{object}}{A_{ref}}. \tag{5.8}$$

Feature 3: *Total extracted area*

The ratio of all extracted objects (total white pixels) is used as the third feature to define the state:

$$WP_{ratio} = \frac{N_{WP}}{N_{TP}}, \tag{5.9}$$

where N_{WP} and N_{TP} are the number of white pixels and the total number of pixels in the image, respectively.

Feature 4: *Number of Objects*

The last parameter to define the state is the number of extracted objects N_O after thresholding.

The number of discrete levels for CE_{Δ}, A_{ratio}, WP_{ratio} and N_O are 8, 6, 3 and 4, respectively and total number of states is $8 \times 6 \times 3 \times 4 = 576$.

Actions

The agent must adjust the threshold value and the size of the structuring element. The action is defined as the adding to/substracting of a specific value $\pm\Delta_g$ to the current threshold. This value is equal to $\pm\frac{1}{10}$ of the difference between the global maximum and minimum gray levels for a coarse adjustment, and $\pm\frac{1}{20}$ for a fine adjustment. For the morphological operator, the agent increases/decreases the size of the structuring element by 1 in the set $\{0, 1, ..., 5\}$.

Reward/Punishment

In order to define an objective reward/punishment, we need a similarity measure between the ideal and the extracted object. A fast method for this purpose is to use CE_Δ (Eq. 5.7) for comparing the objects. For the small values of CE_Δ there is higher similarity and the agent receives high reward values, while for the medium values it will receive a zero reinforcement, and for the large values of CE_Δ it will be punished:

$$r = \begin{cases} +10 & CE_\Delta \leq \epsilon_1, \\ 0 & \epsilon_1 < CE_\Delta < \epsilon_2, \\ -10 & CE_\Delta \geq \epsilon_2, \end{cases} \tag{5.10}$$

where $\epsilon_1 = 0.04$ and $\epsilon_2 = 0.2$ have been set empirically.

5.1.2 Offline Training and Testing New Images

The global RIS was trained during the offline mode by using the reference object in Figure 5.2. The agent explores the parameter space to assign the Q-matrix with appropriate values to reflect the proper action policy for each given state. The values of learning α, and γ, were heuristically set to 0.7 and 0.6, respectively. On a computer with Genuine Intel CPU @ 1.15GH using Matlab 6, the time for training was 32s. After training, the RIS can be used on new images. The object in these images may be rotated and/or translated. Also, the illumination conditions may vary. Other non-similar and/or similar objects may exist in the image as well. The agent must find the appropriate threshold and post-processing parameters such that the reference object can be correctly extracted.

Twenty test images in the four following groups were used in order to verify the segmentation accuracy:

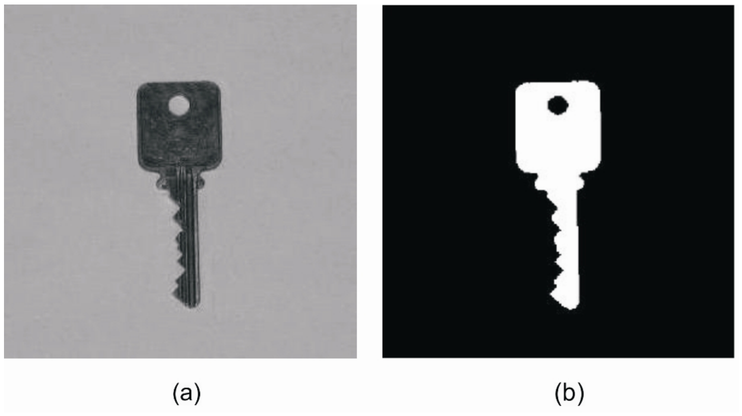

Figure 5.2: (a) Original "key" image. (b) Optimally extracted object.

- **Group A:** Similar and non-similar objects - no reference object.
- **Group B:** Non-similar objects and reference object.
- **Group C:** Non-similar objects - no reference object.
- **Group D:** Similar objects and reference object.

For each group, five images are used. Figure 5.3 and 5.4 show some sample images for each group. In these images, if the reference object exists, it might be rotated and its position changed. Similar objects (different keys) and non-similar objects (pen, eraser, magnifier, coin, etc.) have been added to those images. Also, there is a variation in background illumination. In all cases, the agent can converge to the state indicating a difference of less than 5% to the reference object (whose information is available from training image) and terminate the process in average time $2.9s$. When the agent could not find any match after many iterations (50 for the conducted experiments), it was assumed that there was no reference object in the image.

Table 5.1 shows the summarized results for all images. To show that the RIS system outperforms two of the most commonly used thresholding algorithms [70], with respect to the final recognition rate, we have applied the Kittler and Otsu methods to the images and then used a morphological closing (in each case we used the best result between Kittler and Otsu). The criterion for success requires extracting an object of the same difference to the reference object as used in RIS system ($\leq 5\%$). For the post-processing after the Kittler and Otsu algorithms, we have to use an appropriate size for the structuring element in the morphological operator, while for the RIS system it can automatically be set for each image. The results show that the proposed RIS system is superior to static thresholding.

Table 5.1: Comparison of RIS with the best results of Kittler and Otsu methods.

Image Group	**Results of Kittler and Otsu**		**RIS**	
	No. of Success	No. of Failure	No. of Success	No. of Failure
A	3	2	3	2
B	3	2	4	1
C	4	1	5	0
D	2	3	4	1
Total	**12**	**8**	**16**	**4**
Success Rate	60%		80%	

5.2 RIS for Local Object Extraction

In this section the RIS system will be locally applied on sub-images to extract one object of interest. As was mentioned in Chapter 4, the image will be divided into several sub-images

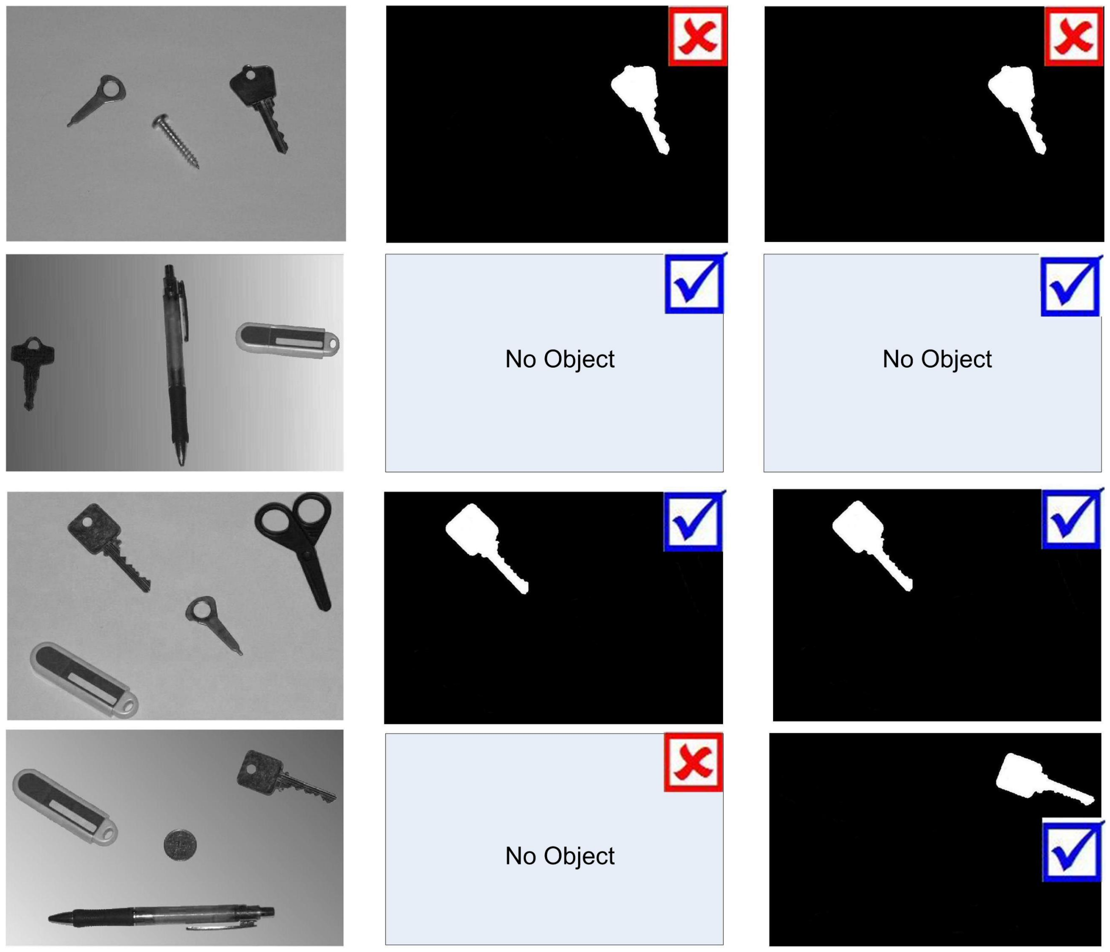

Figure 5.3: Sample images for testing of the RIS system for global object extraction: first, second and third columns show the original images, the best results of Kittler and Otsu methods and the result of RIS, respectively. Rows 1 and 2 are from group A, 3 and 4 from group B. In each image, symbols '✓' and '×' indicate success and failure, respectively.

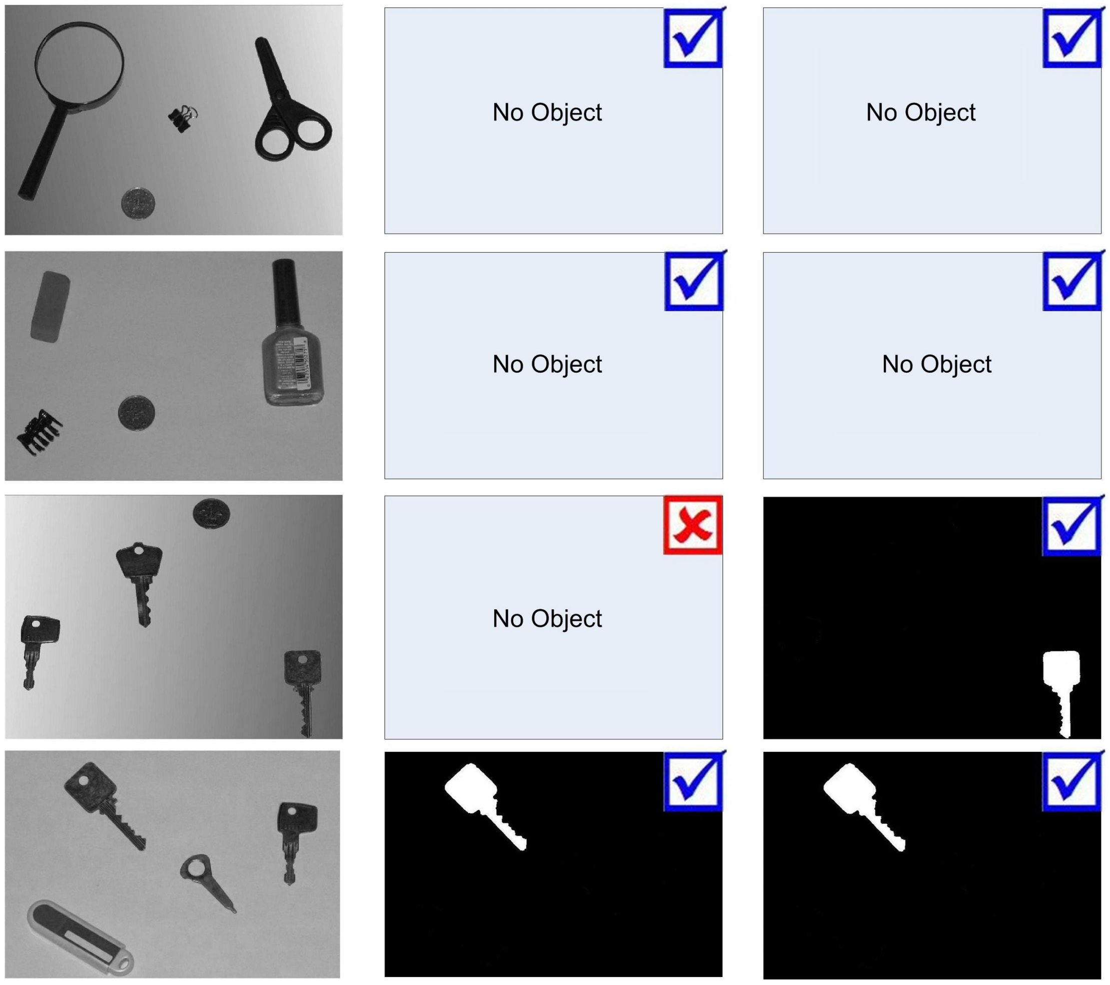

Figure 5.4: Continuation of Figure 5.3. Sample images for testing of the RIS system for global object extraction: first, second and third columns show the original images, the best results of Kittler and Otsu methods and the result of RIS, respectively. Rows 1 and 2 are from group C, 3 and 4 from group D. In each image, symbols '✓' and '×' indicate success and failure, respectively.

of the same size. RIS will reinforce the processing parameters of each sub-image.

5.2.1 Prostate Segmentation in Ultrasound Images

Many applications in medical imaging need to segment one object of interest [43]. Among different image modalities, ultrasound imaging is one of the most widely used technologies for the diagnosis and treatment of such diseases as breast and prostate cancer. These images are the result of reflection, refraction, and deflection of ultrasound beams from different types of tissue with different acoustic impedances [30]. However, poor contrast, speckle, and missing or diffused boundaries make segmentation of these images a challenge.

The prostate segmentation in Trans-Rectal Ultrasound (TRUS) images is a renowned case study [30, 46]. The detection of the prostate boundaries in such images is crucial for cancer diagnosis and classification. However, due to a very low signal-to-noise ratio and presence of shadow, it is difficult to extract the correct boundaries. Further complications arise when the quality of the image is influenced by the type and particular settings of the machine. Although these factors make prostate segmentation a difficult task, ultrasound still remains an important tool for clinical applications and any effort to improve segmentation of these images is highly desirable [1]. Many methods have been introduced to facilitate more accurate segmentation of the prostate boundaries [8, 14, 24, 37, 51, 56, 62, 63, 72, 89]. By studying existing methods, we can observe that they may require many training samples if they rely on learning techniques. Some others need user interactions for each image to determine an initial contour. Also, none of them can improve its performance over time. Considering these factors, a more universal model should require a minimum training data set as well as a reasonable level of user interaction. In addition, continuous online learning is crucial for such applications. In the next section, a RIS system is proposed for prostate segmentation in ultrasound images.

RIS Configuration

A RIS system based on the framework introduced in Chapter 4 is constructed. Due to a low signal-to-noise ratio, a 7×7 median filter is used as pre-processing stage to smooth the image. The pre-processing stage and processing tasks are shown in Figure 5.5. This system can adjust the parameters of a multi-stage segmentation chain by using an RL agent.

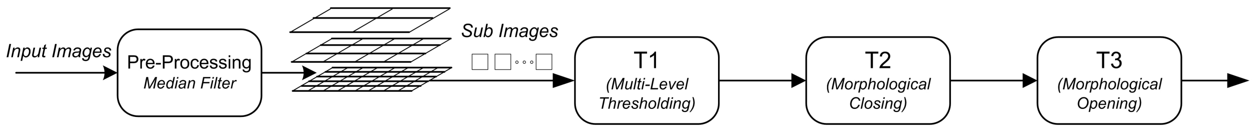

Figure 5.5: RIS pre-processing and processing tasks for ultrasound image segmentation.

We assume that the geometric central point of the prostate (x_c, y_c) in the original image is provided by the user. Figure 5.6 shows this point in a prostate ultrasound image, as well as the assigned sub-images. The RIS system used for segmentation of the prostate in

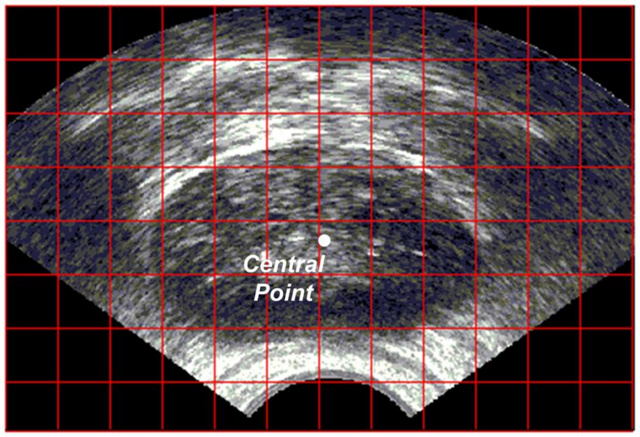

Figure 5.6: A prostate ultrasound image divided in sub-images.

ultrasound images has the following characteristics:

- In terms of processing algorithms, there are three tasks. These tasks are multi-level thresholding with two threshold levels, morphological closing and morphological opening. The parameters of these tasks are reinforced by the RL agent.
- The processing tasks work locally on each individual sub-image.
- For learning, the knowledge acquired during the offline mode is used on the real-time cases, while both objective and subjective online learning can be applied.

Now, a layout is required, where the states are the features that describe the image in various stages, the actions with the capability to change the parameters, and the reward that can be evaluated subjectively or objectively.

States

To generate the states, features of both gray level input and binary output images are used. As the states are locally generated using these features, it is desirable to use those features which do not lead to extremely large state spaces. This is to avoid computational complexity and keep the convergence time short.

According to the potential features explained in Chapter 4, the following features which reflect the local information for each sub-image are extracted to define the states:

Feature 1: *Area*

In each sub-image, the normalized area with respect to the total area of the sub-image is calculated and used as the first feature:

$$A_{norm} = \frac{A_{subimage} - A_{SO}}{A_{subimage}}. \tag{5.11}$$

where $A_{subimage}$ and A_{SO} are the area of the sub-image and the area of its largest object, respectively.

Feature 2: *Compactness*

The compactness Ψ of the largest object after thresholding is defined as:

$$\Psi = \frac{4\pi A_{SO}}{P_{SO}^2}, \tag{5.12}$$

where A_{OS} and P_{OS} are the area and perimeter of the largest object in the sub-image, respectively [25].

Feature 3: *Position Relative to the Centre Point*

By using the geometric center (x_c, y_c) of the prostate given by the user, the relative distance ρ and angle ϕ of the sub-image with respect to the geometric center is adopted as a state parameter:

$$\rho = \left((x_s - x_c)^2 + (y_s - y_c)^2\right)^{\frac{1}{2}}, \tag{5.13}$$

$$\phi = \tan^{-1}\left(\frac{y_s - y_c}{x_s - x_c}\right), \tag{5.14}$$

where x_s and y_s are the coordinates of the center of the current sub-image.

Feature 4: *Gray Level Information*

A histogram prototype on the edge of the prostate can be calculated in the ground-truth images, which shows the gray level distribution in true edgy areas. The histogram distance between each sub-image and the prototype is then calculated and used to define a feature. One of the most popular histogram distances, the χ^2 distance, is selected for this purpose [82]:

$$D_{\chi^2}(h_1, h_2) = \sum_{b=1}^{M_h} \frac{(h_1(b) - h_2(b))^2}{(h_1(b) + h_2(b))}, \tag{5.15}$$

where M_h is the number of gray levels and h_1 and h_2 are the normalized histograms (gray level probability density function) of each sub-image and the prototype, respectively.

Feature 5: *Number of the Objects*

The last feature, used in state definition, is the number of revealed objects, N_O, after the morphological opening.

Actions

To extract the object of interest, the actions are defined to change the control parameters for each sub-image in $Task_1, Task_2$ and $Task_3$. In $Task_1$ sub-images are thresholded using two local levels. Due to disturbances, such as speckle or poor contrast, artifacts exist after thresholding. Therefore, opening and closing, as two morphological operators in $Task_2$ and $Task_3$, are employed to post-process each thresholded sub-image. The RL actions are defined to change the values of multi-level thresholding (two levels) and the size of the structuring elements for each sub-image. The agent changes the assigned local threshold values by a specific amount $\Delta_g = \frac{g_{lmax} - g_{lmin}}{10}$. Using this definition, the values τ_{g_1} and τ_{g_2} among a predefined set $(\tau_1, \tau_2, ..., \tau_n)$, equally spaced between the local maximum gray level g_{lmax} and local minimum gray level g_{lmin}, are taken and all gray values $\tau_{g_1} < g < \tau_{g_2}$ are extracted as object intensity. For the post-processing tasks, the size of the disk shape structuring element for the closing, ν_c, is changed by 1 in the set $\{0, ..., 5\}$, and for the opening, ν_o, by 5 in the set $\{0, ..., 15\}$.

Thus, the set of all actions $\mathcal{A}$ can be presented as follows:

$$\mathcal{A} = \{\tau_{i_1} \pm \Delta_g, \tau_{i_2} \pm \Delta_g\} \cup \{\nu_c\} \cup \{\nu_o\}, \tag{5.16}$$

where $\Delta_g = \tau_{i_1} - \tau_{i_1-1} = \tau_{i_2} - \tau_{i_2-1}(i_1, i_2 \neq 0)$.

Reward/Punishment

After the agent adjusts the thresholding values and the size of the structuring element, it receives a reward or punishment. For each sub-image, the agent must receive the reward for improving the quality of segmented object. The reward/punishment must correctly reflect the goal of the system. Therefore, for each sub-image we use the general form described in 4.3.3, where dissimilarity, D_{i_s}, is the measure for misclassified pixels between the result of the system and the desired output.

Offline Learning Mode

In the offline learning mode, the ground-truth images are available. Figure 5.7 illustrates a typical prostate ultrasound image and its manually segmented version for offline training. To train the agent, twelve manually segmented images from the data set containing eighty

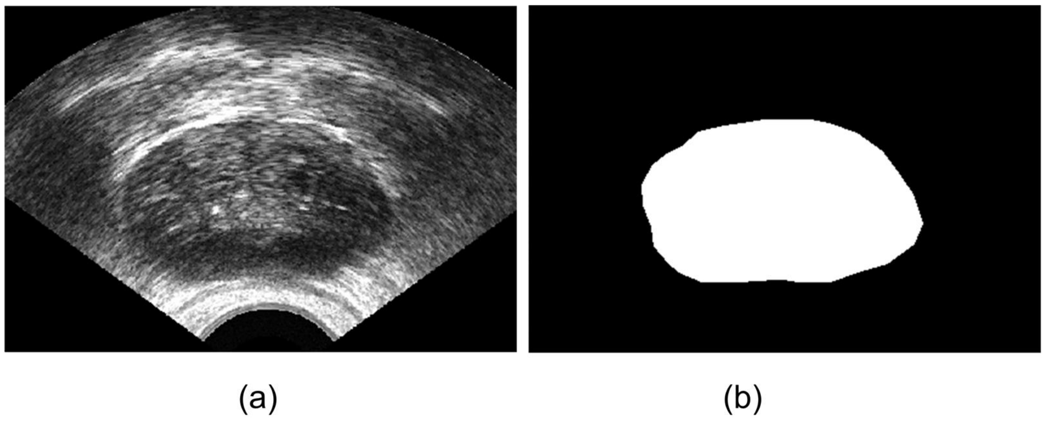

Figure 5.7: (a) Original TRUS image, (b) Manually segmented prostate.

TRUS image slices from five patients were chosen. The ϵ-greedy policy is implemented to explore and exploit the solution space when the RL agent is in the offline mode. The size of the TRUS images is 460×356 pixels. The sizes of $R_I \times C_I$ sub-images are empirically set to $R_I = 11$ and $C_I = 14$ (as the smallest size in the hierarchical model). The number of discrete levels for A_{norm}, Ψ, $\rho\phi$, D_{χ^2}, N_O were 5, 3, 18, 4 and 5, respectively, and total number of states is 5400. The criterion to terminate the process for each sub-image is to reach a pixel difference less than 5% by comparing it with the ground-truth image. The reward is calculated according to $R_1 = 10, R_2 = 0$ (see Eq. 4.3, page 52). The values of learning α, and γ, were heuristically set to 0.8 and 0.6, respectively. On the same computer, the average time for training images was $48s$ per patient set and for test images $7.2s$ per image.

Online Learning Mode

After the offline learning mode, the Q-matrix is filled with appropriate values. To perform the online learning, an objective online evaluation is performed. To implement this evaluation, we apply the matching technique introduced in section 4.5 using the central distance as the signature of the segmented shape. The central distance is calculated and represented as a 2π periodic function to generate the signature. As we have the central point $O(x_c, y_c)$- given by the user- this representation is quite reliable. In this case, an angle θ_C is assigned to a distance d which is the nearest corresponding contour point as follows:

$$d = \left[(x_p - x_c)^2 + (y_p - y_c)^2\right]^{\frac{1}{2}}, \tag{5.17}$$

where (x_p, y_p) are the cartesian coordinates of the pixels located on the border of the prostate. Figure 5.8 shows distance d and angle θ_C for a sample point on the prostate border. As it can be seen, θ_C is the angel between the vertical axis and the line which connects the points (x_c, y_c) and (x_p, y_p). This angle discretely increases to cover the period

$[0, 2\pi]$ and for each angle the corresponding distance d is achieved. Because we use the geometric center of the object, this representation is invariant to translation. Also, we normalize d to make this transformation scale invariant. In addition, because we merely need to detect irregular points in the signature, this method is not sensitive to orientation. Figure 5.8 shows d and θ_C for a sample point on the prostate border.

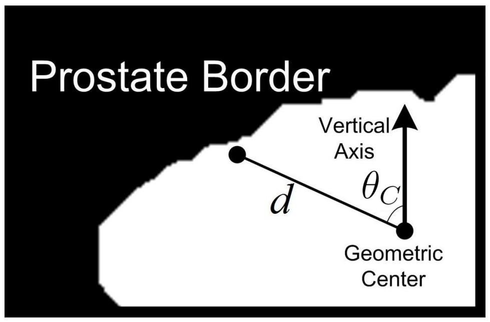

Figure 5.8: d , θ_C for a sample point on the prostate border.

The matching technique can detect those parts which have a large deviation from the regular signature. Figure 5.9 (a) illustrates the irrelevant parts that may be revealed after thresholding. Figure 5.9 (b) shows the points used to create the signature and Figure 5.9 (c) shows the result of applying the matching technique on the signature of the segmented object. The above technique needs to be applied to the entire segmented object. When the matching technique detects irregular parts, the RIS system notes in which sub-image they are located. Finally, online objective reward r^{ON} for the corresponding sub-image and corresponding action is provided. For subjective evaluation, the user considers the results of the RIS system for each image from his/her global point of view and may change some parts manually if he/she is not satisfied. These changes are evaluated as online subjective reward r^{ON}_{SUBJ} for the action of system on corresponding sub-image. The values for objective and subjective online reward are the same as those for offline reward. The RIS system uses

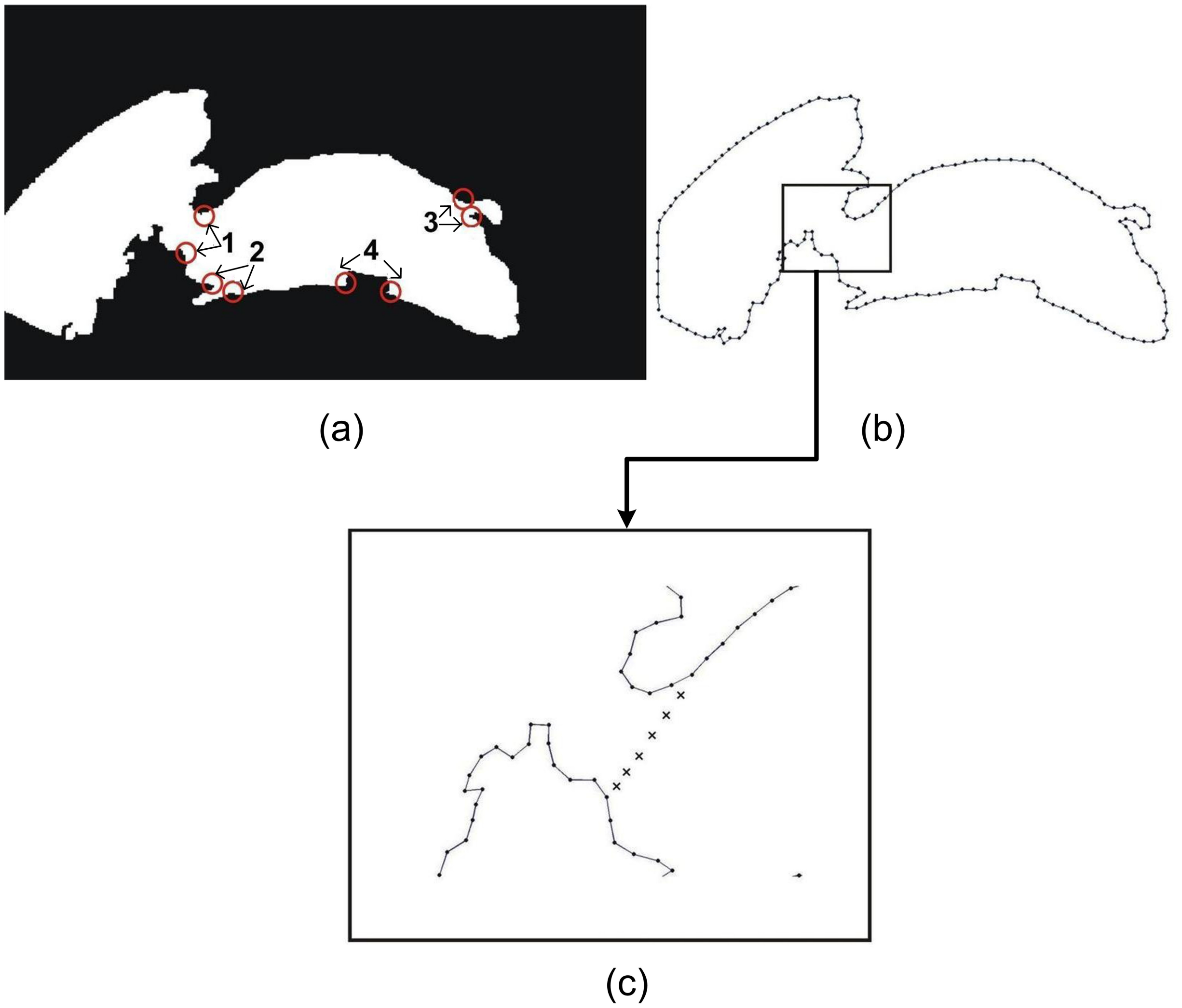

Figure 5.9: Applying the matching technique on the signature of the segmented object. (a) Numbers 1,2 and 3 show three attached parts and number 4 a missing boundary segments, (b) the points on the boundary of the segmented object used to create the object signature, (c) the result of the matching technique on the signature of the segmented object for part number 1 (the irrelevant part can be detected).

the manually corrected result as a new ground-truth image to update its knowledge for future images.

During all of these procedures, the object of interest can be extracted using the position of geometric center given by the user. Standard processes, such as boundary refinement can also be applied on the extracted object to achieve a well-shaped final result.

The method is applied on the remaining sample images. Figure 5.12 displays the results of the proposed approach for 15 sample images.

Quantitative Measurements

For all images, the accuracy of the final segmented object is defined as *Area Overlap*, AO, which is a commonly used metric for this purpose [14]. Using the terms TP (True Positive), FP (False Negative) and FN (False Negative) for the classified pixels, the area overlap is calculated as follows:

$$AO = \frac{TP}{TP + FN} \times \%100. \tag{5.18}$$

Table 5.2 shows the results for training images (offline RIS) and Table 5.3 for test images (online RIS). Because in offline RIS, there are a few number of training samples (2 or 3) for each patient, the standard deviations are not shown in Table 5.2. Considering the segmented TRUS images in terms of the visual appearance and the analysis presented in [14, 46], the RIS system shows very promising results.

To understand the concept of learning in the proposed RIS system, the AO curve for each patient is depicted in Figures 5.13 - 5.17. As can be seen in these Figures, the AO curves have an increasing trend in the online mode. This shows that the system is permanently learning by processing more samples.

Figure 5.18 shows the area overlap accuracy for patient 1 in more detail. For each accuracy point, the corresponding image is also shown. As a matter of fact, the drop in the

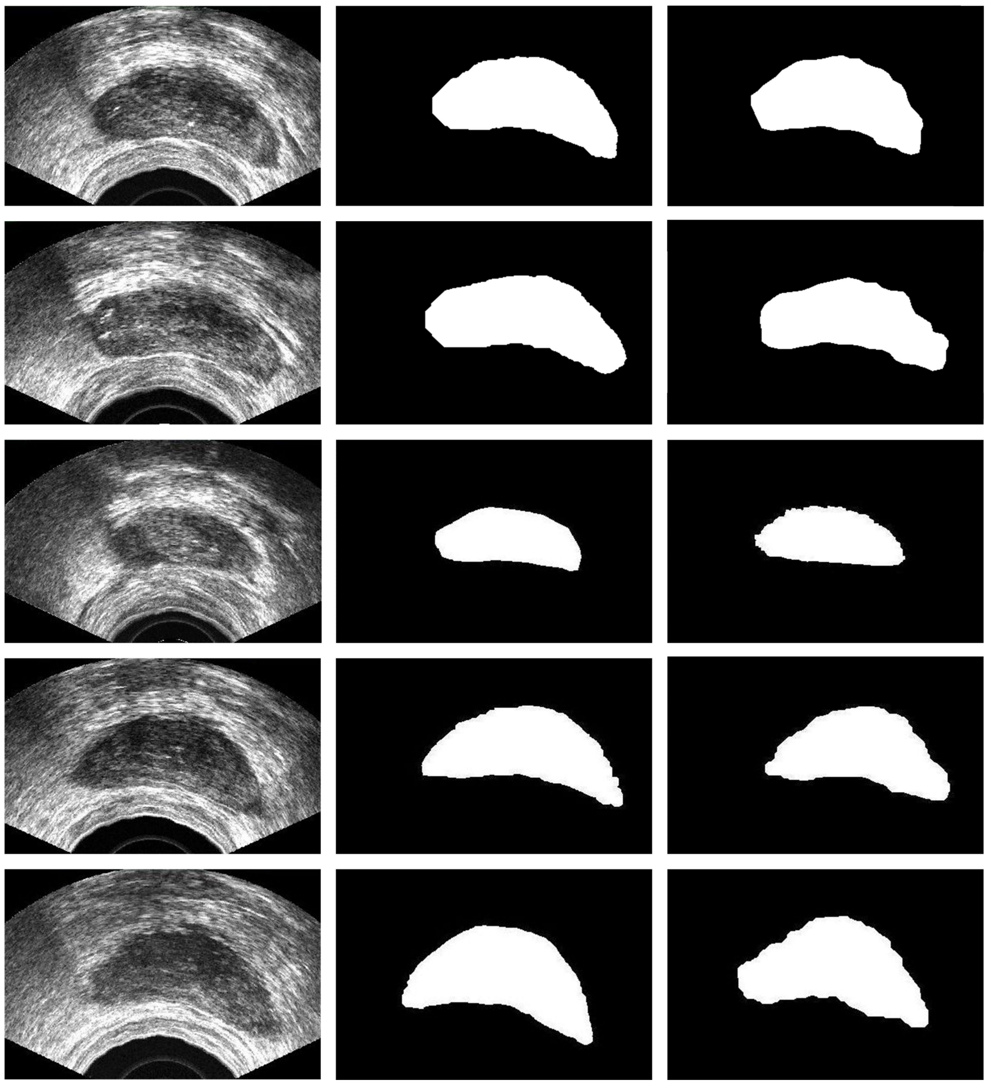

Figure 5.10: Image $1-5$ for demonstration of the RIS system for prostate segmentation in TRUS images. The first, second and third columns are original, manually segmented and result of the proposed RIS system, respectively.

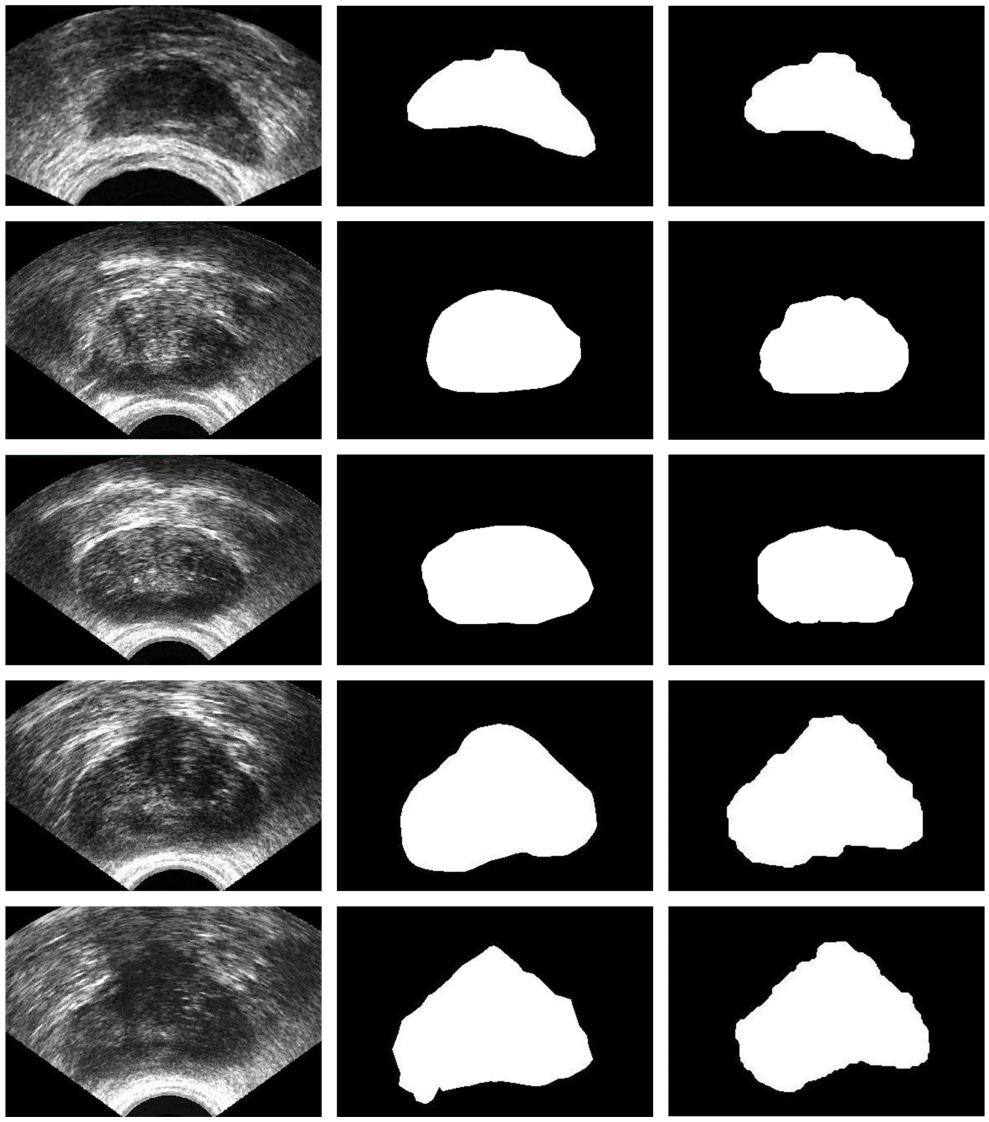

Figure 5.11: Image $6 - 10$ for demonstration of the RIS system for prostate segmentation in TRUS images. The first, second and third columns are original, manually segmented and result of the proposed RIS system, respectively.

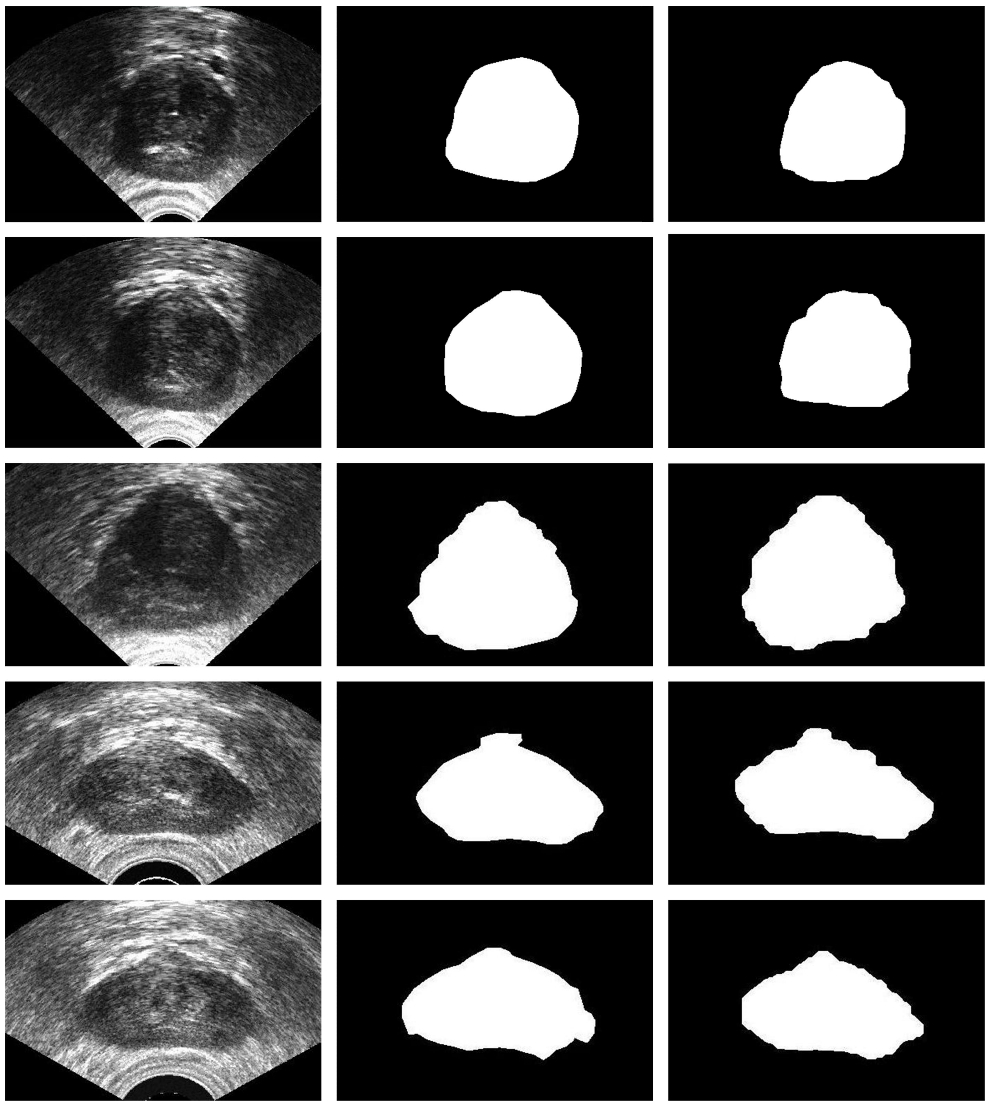

Figure 5.12: Image $11 - 15$ for demonstration of the RIS system for prostate segmentation in TRUS images. The first, second and third columns are original, manually segmented and result of the proposed RIS system, respectively.

Table 5.2: The average area overlap AO between the segmented and the ground-truth images for the training images of patients $1-5$ (offline RIS).

	$\boldsymbol{AO}(\%)$ Patient 1	$\boldsymbol{AO}(\%)$ Patient 2	$\boldsymbol{AO}(\%)$ Patient 3	$\boldsymbol{AO}(\%)$ Patient 4	$\boldsymbol{AO}(\%)$ Patient 5
μ	**93.56**	**95.30**	**94.07**	**94.58**	**93.83**

Table 5.3: The means μ and standard deviations σ of area overlap AO between the segmented and the ground-truth images for test images of patients 1-5 (online RIS).

	$\boldsymbol{AO}(\%)$ Patient 1	$\boldsymbol{AO}(\%)$ Patient 2	$\boldsymbol{AO}(\%)$ Patient 3	$\boldsymbol{AO}(\%)$ Patient 4	$\boldsymbol{AO}(\%)$ Patient 5
μ	**90.74**	**91.05**	**89.07**	**92.26**	**91.12**
σ	**1.04**	**1.12**	**1.06**	**0.91**	**0.72**

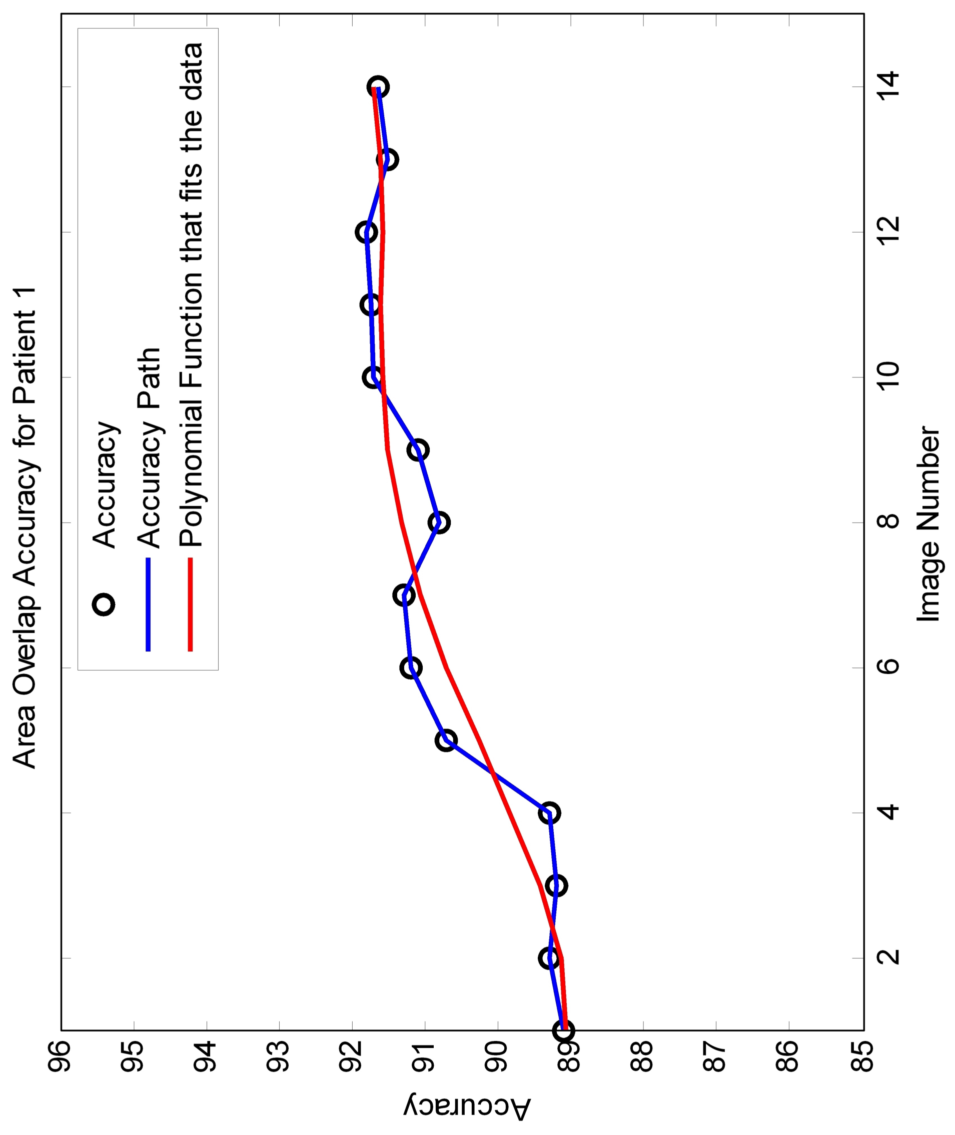

Figure 5.13: Area overlap accuracy for patient 1 in online learning mode.

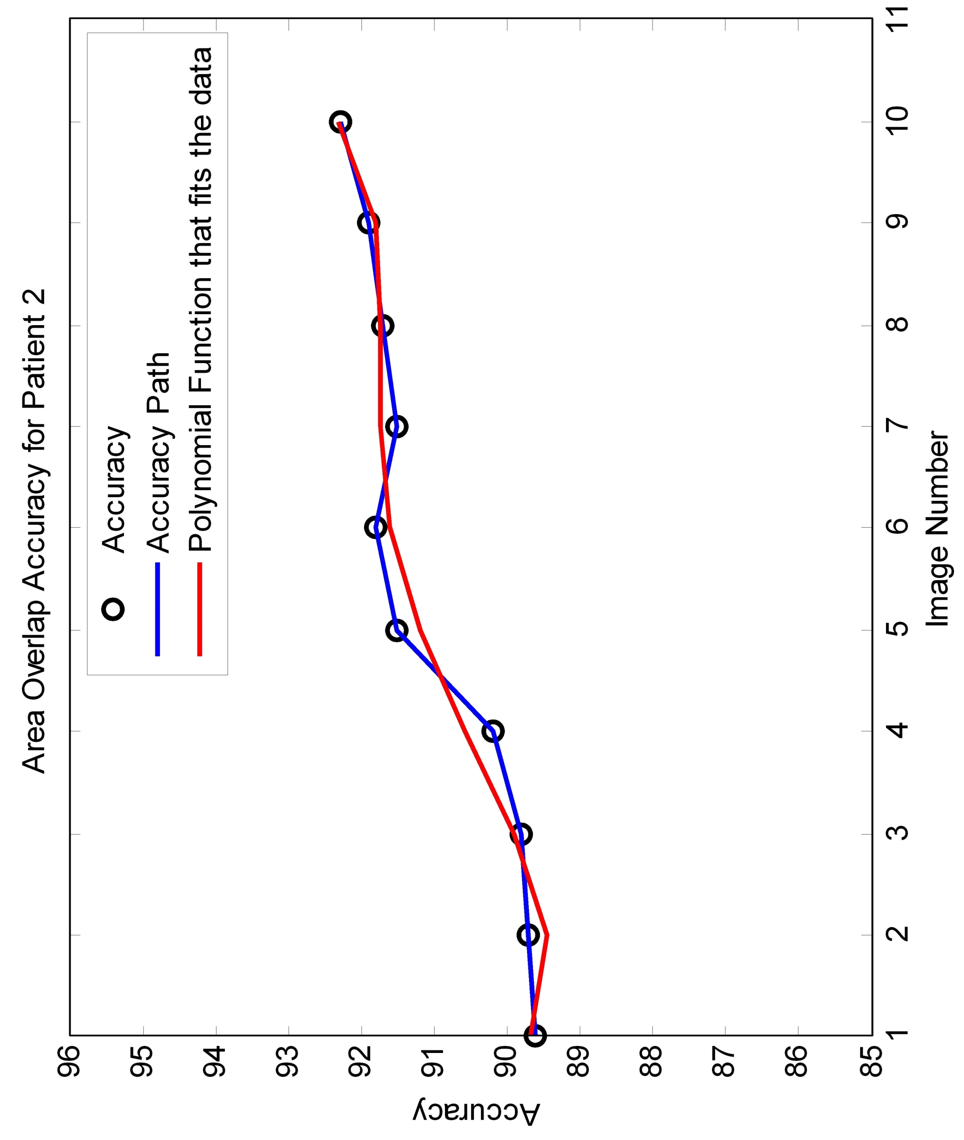

Figure 5.14: Area overlap accuracy for patient 2 in online learning mode.

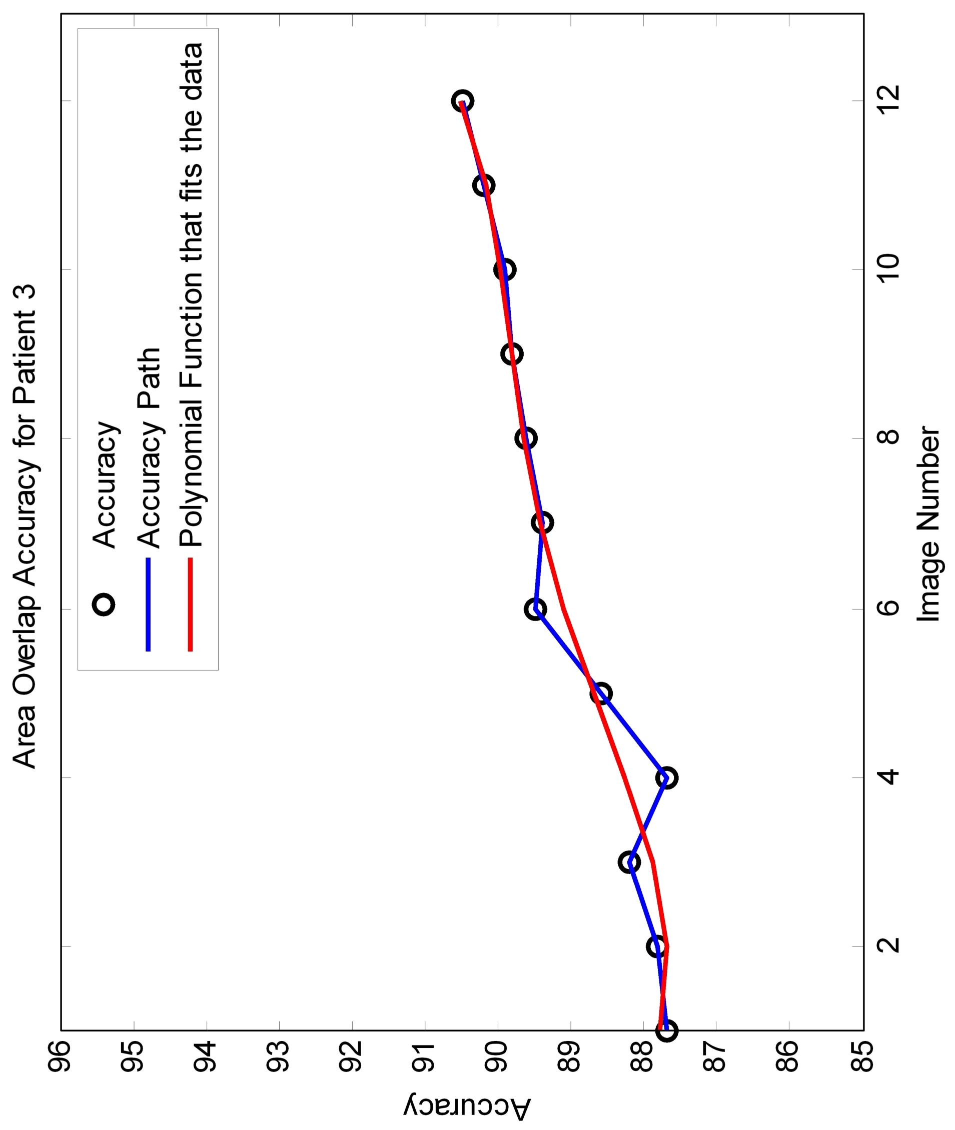

Figure 5.15: Area overlap accuracy for patient 3 in online learning mode.

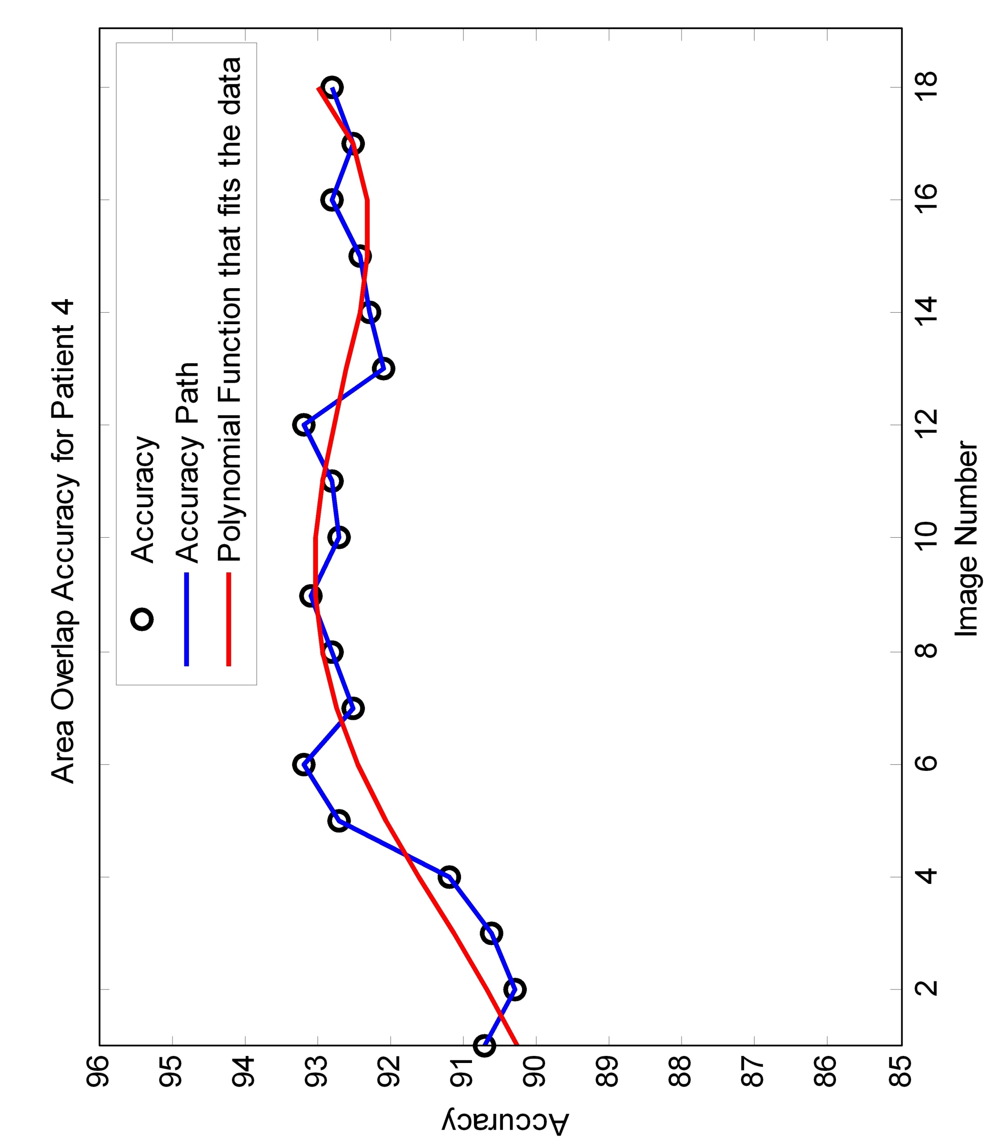

Figure 5.16: Area overlap accuracy for patient 4 in online learning mode.

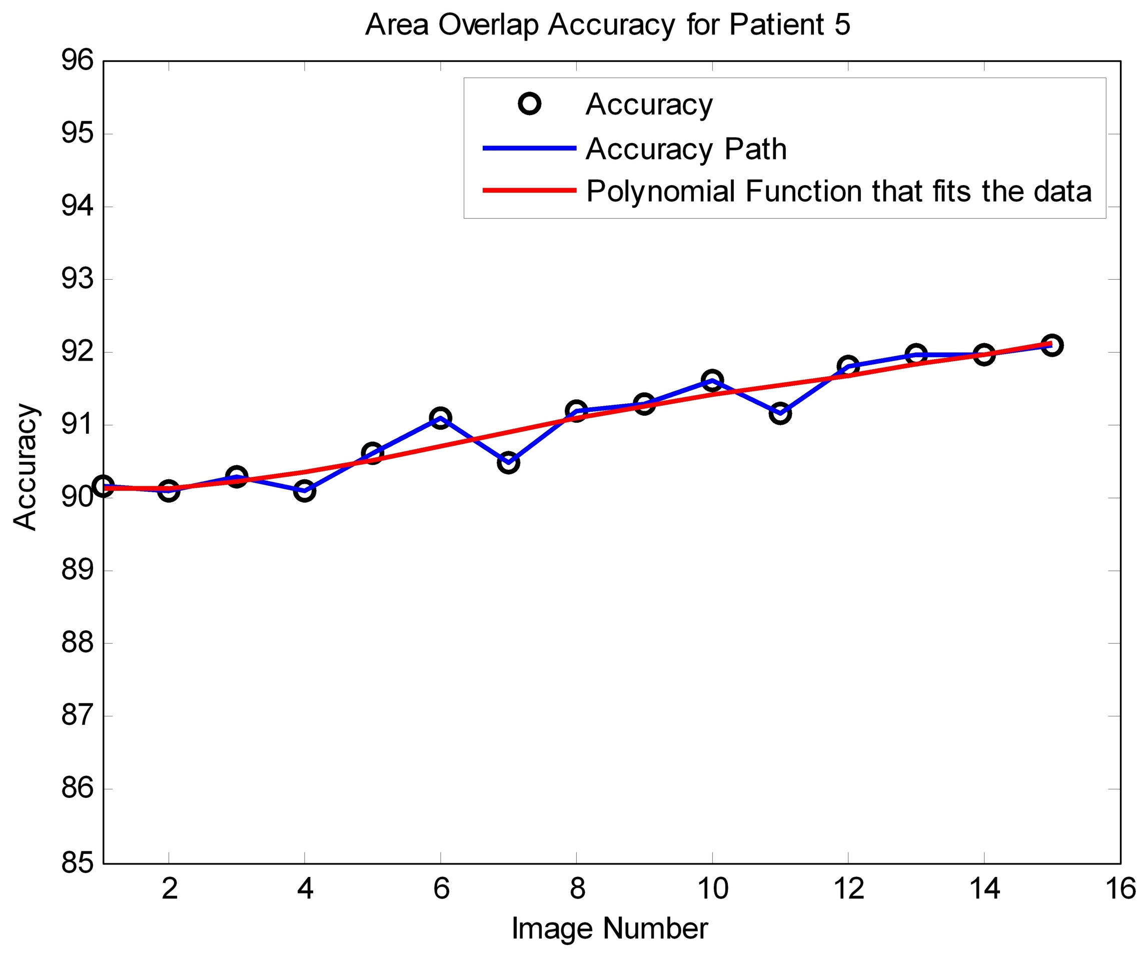

Figure 5.17: Area overlap accuracy for patient 5 in online learning mode.

curve occurs when the shape of the prostate changes or the quality of the image decreases. But after interacting with the new cases, the RIS system learns the new circumstances and adapts itself. As a result, accuracy again increases. This clearly demonstrates the online capability of the RIS system.

Opposition-Based RIS System

In the previous section, a conventional RL algorithm was used for segmentation of prostate. As mentioned previously, a potential obstacle when we applying RL agents to image-based applications, is the large number of state-action pairs involved. Therefore, we need to speed up the learning process to increase the practicality of the RIS system. Opposition-based learning is one method that can be applied for this purpose.

In order to make additional updates, the RL agent has to know how to find "opposite actions" and "opposite states". In the prostate segmentation, this is only performed for actions. For example, if the action is to increase the threshold value, the opposite action is defined as decreasing it. When employing opposition-based learning, the training time is reduced during the exploration mode. This is because the Q-matrix is filled more rapidly using extra opposition-based updates. The system structure is the same as designed previously to segment the prostate TRUS images. This is shown in figure 5.19 where the new unit has been added to the RIS system.

Table 5.4 shows the average Learning Time Reduction (LTR%) with respect to the conventional Q-Learning used in RIS.

Comparison with Bayesian Classifier

Even though a comparison with other techniques that do not possess any online learning capabilities does not seem to be fair, it would be desirable nevertheless to compare the

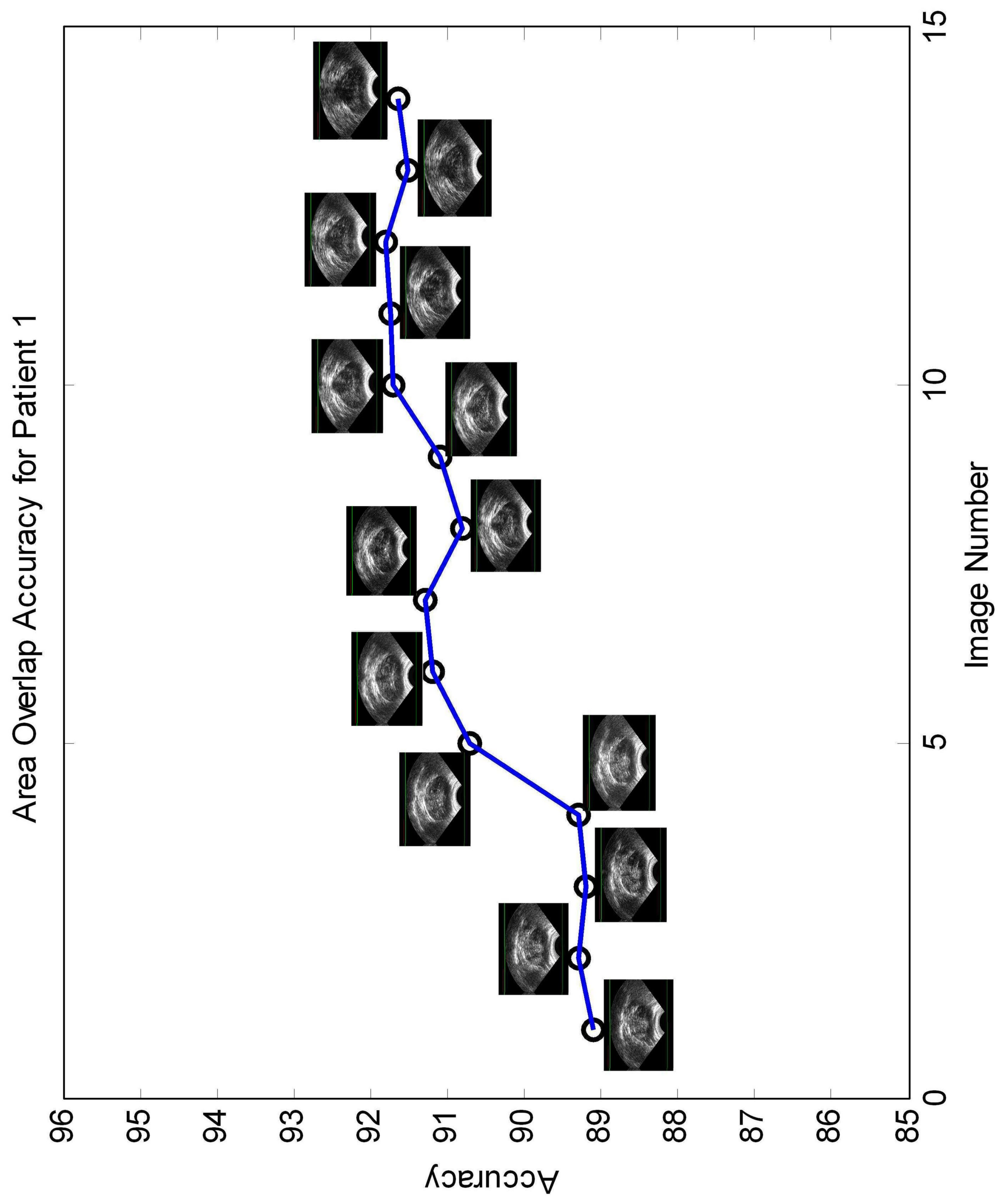

Figure 5.18: Area overlap accuracy for patient 1 including the corresponding images.

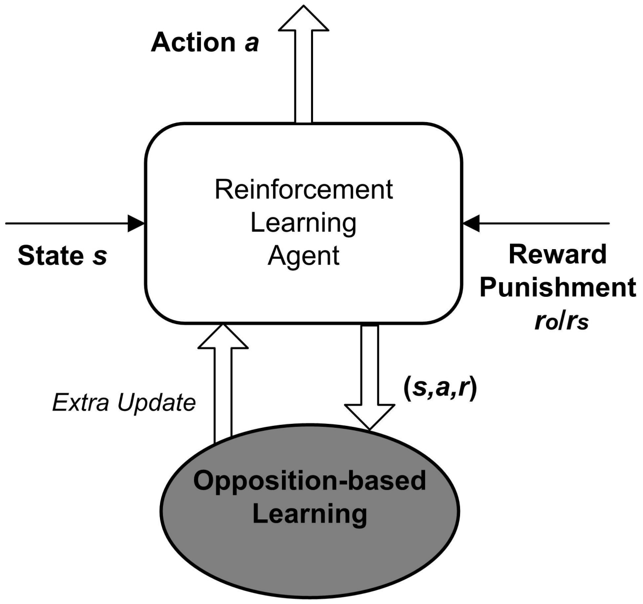
Action *a*
Reinforcement Learning Agent
State *s*
Reward Punishment *ro*/*rs*
Extra Update
(*s*,*a*,*r*)
Opposition-based Learning

Figure 5.19: Opposition-based RIS for TRUS images (see Figure 4.4, page 54).

Table 5.4: Learning Time Reduction (LTR%) for RIS using opposition-based learning comparing to the strandard Q-learning for patients $1-5$.

	Patient 1	Patient 2	Patient 3	Patient 4	Patient 5
LTR%	17.2%	18.2%	19%	16.7%	21.5%
$\mu = 18.52$			$\sigma = 1.88$		

segmentation results with existing methods. Hence, a Naive Bayesian (NB) classification approach to prostate ultrasound image segmentation was investigated. There are two possible classifications for a pixel in an ultrasound image: prostate or not prostate. NB requires a set of training data and utilizes a number of attributes in order to determine the class for a given instance. For new images, NB will determine each pixel class based on the attribute values for that pixel and probabilities associated with those attribute values in the training data set.

Feature Analysis

The performance of a number of features has been analyzed and the best ones were selected to achieve the maximum possible accuracy for such a classifier. The features investigated are pixel position, gray level intensity, and texture based on Gabor filter, local range, local standard deviation and local entropy. These are explained as follows:

- Pixel Position - The absolute position of the pixel.
- Gray Level Intensity - The gray level value of a pixel.
- Gabor Features - Gabor filters can be applied to identify texture patterns for a given frequency in a given orientation. For this study, 6 orientations ($0°$ to $150°$ in $30°$ increments) were used.
- Local Range - The difference between the maximum and minimum intensity values around a pixel within a given window (3×3 and 5×5 windows).
- Local Standard Deviation - The standard deviation of the intensity values around a pixel within a given window.
- Local Entropy - Entropy is found for a given neighbourhood around a pixel.

Experimental Results

The data set consisting of 80 images was randomly divided into 70% for training and 30% for validation. When we perform this approach on the data set, the experimental results show that the combination of position, gray level intensity and Gabor filters in the 3^{rd}, 4^{th} and 5^{th} orientations (corresponding to 60°, 90° and 120°, respectively) result in the largest average area overlap. This area overlap results in accuracy of about 75%.

For this classifier, it can be seen that there is no significant performance change as a median filter is added for smoothing the image. Using a 9×9 median filter, the average area overlap accuracy remains about the same at 75%. An experimental study was also performed where the original data set was categorized by a general prostate shape (walnut- or kidney-shaped). The purpose here was to observe performance changes when images with similar contents are used. In this case, individual walnut- and kidney-shaped image data sets performed better than the data set containing all images. The highest area overlap accuracy is found to be about 84%, using 9×9 median filtered images with position and Gabor filter attributes. The proposed RIS system clearly outperforms the Bayesian classifier in all cases, even when the classifier uses the categorization of the image set. We have to note that for the Bayesian classifier, 70% of the images were used for training, while for RIS this amount was 15% which is also a significant advantage.

Table 5.5 represents the above experimental results. As can be seen, the proposed RIS system possesses significantly better accuracy.

Another case where the application of the local RIS system is utilized, is the extraction of a particular object from an image which is explained in detail in Appendix C.

Table 5.5: Area overlap *AO* for Bayesian classifier and RIS.

	Bayesian for All Images	Bayesian for Categorized Images	RIS System on Test Images
μ	74.69%	84.14%	90.84%
σ	0.02	0.02	0.97

5.3 Chapter Summary

In this chapter, the proposed RIS system was applied on three different cases. These applications use the general framework as proposed in Chapter 4, in three different configurations to show the potential of the system. The first application was RIS for global object extraction, which uses the knowledge obtained from online learning only as a lookup table with no online learning. The second application was RIS for local object segmentation of the prostate in ultrasound images, with online learning and both objective and subjective online learning. The last application was the extraction of a particular object in the image that uses offline and objective online learning, with no subjective feedback. It was demonstrated that the RIS system could extract the object of interest more effectively than conventional methods.

This system is specifically useful for the applications where images have widely the same object and background characteristics and an expert supervises the performance of the system. Medical image segmentation is the most apparent application for RIS.

Chapter 6

Conclusions and Future Perspectives

This chapter explains the original contributions of this research, as well as pointing out how this work opens new directions for future research.

6.1 Research Contributions

This thesis introduced Reinforced Image Segmentation (RIS) of one object of interest. Determining the optimal parameters for multi-task segmentation processes is a challenging task, as both the amount and nature of information can limit attempts to solve such problems. This thesis has proposed a reinforcement learning framework to segment images containing one object of interest. In this system, we view parameter selection of a multi-task segmentation procedure as a decision process over time, where the experience from past decisions affects future decisions.

The techniques used for segmentation vary depending on the particular situation and the specifications of the problem at hand. In methods which rely on a learning process, the lack of a sufficient number of training samples is usually an obstacle, especially when the samples are being manually prepared by an expert. The performance of other methods may suffer from frequent user interactions required to determine the critical segmentation parameters. Also, none of the existing methods uses online (permanent) feedback from the user in order to evaluate the generated results. Considering the above factors, a new multi-stage image segmentation method based on reinforcement learning was introduced as the main contribution of this research:

- **Structure of the System:** A thorough study to construct a general framework for the targeted system was performed. In this framework, called Reinforced Image Segmentation (RIS) system, the agent can choose the appropriate parameter values for different processing tasks based on its accumulated knowledge. The aim of this system is to identify one object of interest and to separate it from the background. The potential states, actions and rewards have been introduced and their characteristics explained. The application of this system has been successfully demonstrated by finding the appropriate parameters for image processing tasks in different segmentation applications.

- **Offline Learning Mode:** In this mode, the approach starts with a limited number of training samples. Learning occurs based on interactions with an offline simulation environment using the ground-truth images. The agent takes specific actions, such as changing the tasks parameters, to change the quality of the segmented image.

- **Online Learning Mode:** The user knowledge must be incorporated to increase the segmentation capabilities of the system. In this mode, the ground truth-based reward is replaced by rewards based on geometrical properties of the object, as well as the user feedback. The online mode guarantees that the system is continuously trained and can increase its accuracy the more the user works with it.

The experiments have successfully shown that the system can start its work from a limited number of training samples, and increase its accuracy by online learning as the user works with the system.

6.2 Future Research

The work presented in this thesis is an attempt to combine two broad research areas, dynamic learning systems and image processing. This research has provided a new concept for the application of reinforcement learning as a dynamic solution to image segmentation, which is inherently static if single images are processed. The thesis highlights some important aspects in this regard and perhaps raises new questions. At this point, ideas for future research related to this work can be outlined as follows:

- **Feature selection:** State definition is a crucial step in the RIS system. For both input gray level and output binary images, more information from different classes of features can be added. It is expected that this additional information will increase the computational complexity by forming a larger state space. An appropriate approximation method may be employed to reduce this problem.

- **Extension to other image-based applications:** The RIS system could be the basis for other parameter control applications such as contrast enhancement or noise filtering. Also it could be used to segment "several" objects in an image.

- **Applications:** From an application point of view, promising results of this work lead to the applicability of the proposed RIS system to other segmentation problems. This can be performed in different disciplines, especially medical imaging applications such as breast ultrasound or brain MR images.

Appendix A

Policy Estimation

In this appendix, we briefly review two existing schemes, Dynamic Programming (DP) and Monte Carlo (MC) methods, and then introduce a third family of algorithms, Temporal-Difference (TD) methods, which form a bridge between the first two.

A.1 Dynamic Programming

Dynamic programming must know the complete and accurate model of environment as a Markov decision process. With this assumption, we can use DP to solve the reinforcement learning problems. To search for useful policies, the value function must be calculated. In fact, using a value function is one of the most important parts of reinforcement learning. Considering $\mathcal{S}$ as a finite set of states, such that $s \in \mathcal{S}$, $\mathcal{A}(s)$ as a finite set for actions, and the environment as a finite MDP, the next state can be predicted by using equation A.1. The next reward will be predicted using equation A.2, where $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$, and $s^{'} \in \mathcal{S}^{+}$ ($\mathcal{S}^{+}$ is $\mathcal{S}$ plus a terminal state if the problem is episodic) [79].

$$P^{a}_{ss^{'}} = Pr\{s_{t+1} = s^{'} | s_t = s, a_t = a\}, \tag{A.1}$$

$$R^a_{ss'} = E\{r_{t+1}|a_t = a, s_t = s, s_{t+1} = s'\}. \tag{A.2}$$

The next step is finding the value function, V^* or Q^*, which satisfies the Bellman optimality presented in equations A.3 and A.4 where $\forall s \in \mathcal{S}, a \in \mathcal{A}(s)$, and $s' \in \mathcal{S}^+$.

$$V^*(s) = \max_a E\{r_{t+1} + \gamma V^*(s_{t+1})|s_t = s, a_t = a\} = \max_a \sum_{s'} P^a_{ss'}[R^a_{ss'} + \gamma V^*(s')], \tag{A.3}$$

$$\begin{aligned} &Q^*(s,a) = E\{r_{t+1} + \gamma \max_{a'} Q^*(s_{t+1}, a')|s_t = s, a_t = a\} = \\ &\sum_{s'} P^a_{ss'}[R^a_{ss'} + \gamma \max_{a'} Q^*(s', a')]. \end{aligned} \tag{A.4}$$

Considering an arbitrary policy π,the state value function V^π can be computed for all $s \in \mathcal{S}$ using equation A.5, where $\gamma < 1$. $Pr^\pi(s,a)$ is the probability of taking action a in state s using policy π [79].

$$\begin{aligned} &V^\pi(s) = E_\pi\{r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + ...|s_t = s\} = E_\pi\{r_{t+1} + \gamma V^\pi(s_{t+1})|s_t = s\} = \\ &\sum_a Pr^\pi(s,a) \sum_{s'} P^a_{ss'}[R^a_{ss'} + \gamma V^\pi(s')] \end{aligned} \tag{A.5}$$

The iterative policy evaluation is presented in the Figure A.1 algorithm. To implement a program, two arrays are defined, one for old value function and one for new. To terminate the algorithm the stopping condition may be defined such that $\max_{s \in \mathcal{S}} |V_{k+1}(s) - V_k(s)|$ be sufficiently small. To present a better policy the equation A.6 will be calculated [79]:

$$Q^\pi(s,a) = E_\pi\{r_{t+1} + \gamma V^\pi(s_{t+1})|s_t = s, a_t = a\} = \sum_{s'} P^a_{ss'}[R^a_{ss'} + \gamma V^\pi(s')]. \tag{A.6}$$

Policy improvement create a new policy by making it greedy or near greedy, yielding equation A.3. The sequence of improving the policies continues until it converges on

```
Input $\pi$, the policy to be evaluated
Initialize $V(s) = 0$, for all $s \in \mathcal{S}^+$
Repeat
  $\triangle \leftarrow 0$
  For each $s \in \mathcal{S}$ :
    $v \leftarrow V(s)$
    $V(s) \leftarrow \sum_{a} Pr^{\pi}(s,a) \sum_{s^{'}} P^{a}_{ss^{'}} \ [R^{a}_{ss^{'}} + \gamma V(s^{'})]$
    $\triangle \leftarrow max(\triangle, |v - V(s)|$
until $\triangle < \theta$ (a small positive number)
output $V \approx V^{\pi}$
```

Figure A.1: Iterative policy evaluation [79].

an optimal policy and optimal value function (finite MDP has finite policies) in finite iterations. The algorithm for finding the optimal policy (policy iteration) is given in Figure A.2 for V^*. To prevent computational costs, resulting from policy evaluation, we can use value iteration by tuning the Bellman optimality equation into an updated rule (see algorithm presented in Figure A.3). One drawback of DP is operations over the entire state of the MDP, and if the state is huge then it is computationally expensive. A synchronous DP algorithm backs up the values of other available states. For the sake of convergence, these algorithms can not ignore any state. Asynchronous DP gives flexibility in selecting states and makes it easier to intermix computation with real time interaction [79].

A.2 Monte Carlo Algorithm

In contrast to dynamic programming, the Monte Carlo (MC) method is model-free but it is not a suitable technique for step-by-step incremental computation. In this method, the environment is sampled by executing the current policy and only the true observed reward is used for value updates. In fact, in the MC method the agent learns online or from simulated interaction with an environment and gains a sample sequence of states, actions, and reward. This way of solving an RL problem based on averaging sample returns, (in particular by defining Monte Carlo only for episodic tasks in such a way that experience is divided into episodes that will terminate, then the value will be estimated and policies will be changed). If Monte Carlo gathers online experience, it will require no prior knowledge of the environment's dynamics. If it learns from simulated experience, however, the model needs only generated sample transitions, not the complete probability distributions of all possible transitions.

```
1. Initialization
$V(s) \in R$, and $\pi(s) \in \mathcal{A}(s)$ arbitrary ($\forall s \in \mathcal{S}$)
2. Policy Evaluation
Repeat
    $\Delta \leftarrow 0$
    For each $s \in \mathcal{S}$:
    $v \leftarrow V(s)$
    $V(s) \leftarrow \sum_{s'} P_{ss'}^{\pi(s)}[R_{ss'}^{\pi(s)} + \gamma V(s')]$
    $\triangle \leftarrow max(\triangle, |v - V(s)|)$
until $\triangle < \theta$ (a small positive number)
3.Policy Improvement
$Policy - stable \leftarrow$ true
For each $s \in \mathcal{S}$:
    $b \leftarrow \pi(s)$
    $\pi(s) \leftarrow \operatorname{argmax}_a \sum_{s'} P_{ss'}^{a}[R_{ss'}^{a} + \gamma V(s')]$
    If $b \neq \pi(s)$, then $policy - stable \leftarrow$ false
If $policy - stable$, then stop; else go to 2
```

Figure A.2: Policy iteration for V^* [79].

```
Initialize $V$ arbitrary, e.g., $V(s) = 0$, for $\forall s \in \mathcal{S}^+$
Repeat
    $\triangle \leftarrow 0$
    For each $s \in \mathcal{S}$:
        $v \leftarrow V(s)$
        $V(s) \leftarrow \max_a \sum_{s'} P^a_{ss'}[R^a_{ss'} + \gamma V(s')]$
        $\triangle \leftarrow max(\triangle, |v - V(s)|)$
Until $\triangle < \theta$ (a small positive number)
output a deterministic policy, $\pi$, such that $\pi(s)=\text{argmax}_a \sum_{s'} P^a_{ss'}[R^a_{ss'} + \gamma V(s')]$
```

Figure A.3: Value iteration [77].

To estimate the value of state (expected return or expected cumulative future discount reward starting from that state) the agent has to average the returns observed after visiting each state. After a while the average should converge to the expected value.

There are two techniques in the Monte Carlo method, every-visit and first-visit. In the every-visit Monte Carlo method, $V^\pi(s)$ is estimated as the average of the returns following all the visits to s in a set of episodes. In the first-visit in Monte Carlo method, the average of the returns (following first visit to s) will be calculated. First-visit and every-visit methods converge to $V^\pi(s)$. The algorithm of the first-visit in Monte Carlo method is presented in Figure A.4. In cases where the model is not available, it would be useful to estimate the value of each action for choosing the appropriate policy. The policy evaluation problem is estimating $Q^\pi(s,a)$, the expected return starting in state s and taking action a and following policy π. There are also two different methods for cases in which the

```
Initialize:
    $\pi \leftarrow$ policy to be evaluated
    $V \leftarrow$ an arbitrary state-value function
    $Returns(s) \leftarrow$ an empty list, for all $s \in \mathcal{S}$
Repeat forever:
    (a) Generate an episode using $\pi$
    (b) For each sate $s$ appearing in the episode:
            $R \leftarrow$ return following the first occurrence of $s$
            Append $R$ to $Returns(s)$
            $V(s) \leftarrow$ average $(Returns(s))$
```

Figure A.4: Algorithm of first-visit Monte Carlo method [79].

model is not available, every-visit and first-visit. Every-visit method estimates the value of a state-action pair as the average of returns followed by visits to the state in which the action was selected. First-visit Monte Carlo calculates the average of returns following the first time in each episode that the state was visited and an action was selected. These two methods converge to the true expected values as the number of visits to each state-action pair approaches infinity [79].

The problem is that many relevant state-action pairs may never be visited. The most useful method for assuring that all state-action pairs will be selected is to consider the stochastic policies with a nonzero probability of selecting all actions.

Policy improvement is making the policy greedy, with respect to the current value function which for each $s \in \mathcal{S}$, chooses an action with maximal Q-value (See equation A.7):

$$\pi(s) = arg \max_a Q(s, a). \tag{A.7}$$

To avoid infinite number of episodes, we may complete policy evaluation before returning to policy improvement. The other way is measurements and assumptions are made to gain bounds on the magnitude and probability error in the estimates, and taking steps during each policy evaluation to assure that these bounds are sufficiently small. In the Monte Carlo control algorithm, after each episode the observed returns will be used and then policy will be improved for all states visited in the episode. A complete algorithm of Monte Carlo is presented in Figure A.5.

A.3 Temporal-Difference Learning

The main idea behind the Temporal Difference method is that the prediction is calculated as a sum of discounted rewards. In this method, values of all intermediate states are used as

```
Initialize:, for all $s \in \mathcal{S}, a \in \mathcal{A}(s)$:
        $Q(s,a) \leftarrow$ arbitrary
        $\pi(s) \leftarrow$ arbitrary
        $Returns(s,a) \leftarrow$ empty list
Repeat forever:
        (a) Generate an episode using exploring starts and $\pi$
        (b) For each pair $s, a$ appearing in the episode:
              $R \leftarrow$ return following the first occurrence of $s, a$
              Append $R$ to $returns(s,a)$
              $Q(s,a) \leftarrow$ average($Returns(s,a)$)
        (c) For each $s$ in the episode:
              $\pi(s) \leftarrow \operatorname*{argmax}_a Q(s,a)$
```

Figure A.5: Monte Carlo control algorithm assuming exploring starts [79].

estimating from estimates (bootstrapping). Temporal-difference learning is a combination of monte carlo and dynamic programming ideas. It does not require a model and can learn directly from the environment. It also updates estimates based on other learned estimates without waiting for the final outcome. To find an optimal policy, these three methods (DP, MC, and TD) use some common policy iteration but their approaches to the prediction problem are different. If at time t a nonterminal state s_t is visited, TD estimates $V(s_t)$ are based on what happens after that visit. The TD method waits until the next time step $(t+1)$ to determine the increment to $V(s_t)$. This is in contrast to Monte Carlo, which waits until the end of the episode. The simplest TD method (TD(0)) is presented in equation A.8 where α is constant and $r_{t+1} + \gamma V_t(s_{t+1})$ is target [77]:

$$V(s_t) \leftarrow V(s_t) + \alpha[r_{t+1} + \gamma V(s_{t+1}) - V(s_t)]. \tag{A.8}$$

The tabular TD(0) for estimating V^π presented in Figure A.6. The TD method is an incremental online learning and does not wait until the end of the episode; therefore, it is faster. For any fixed policy π, the TD algorithms converge to V^π with probability 1 if relations in equation A.9 hold:

$$\sum_{k=1}^{\infty} \alpha_k(a) = \infty, \quad \text{and} \quad \sum_{k=1}^{\infty} \alpha_k^2(a) < \infty \tag{A.9}$$

where $\alpha_k(a)$ is the step-size parameter and a is the action. It has been shown that whether TD methods converge faster than constant-α MC methods in practice [79].

Sarsa

Sarsa is on-policy TD control. For an on-policy method the $Q^\pi(s, a)$ must be estimated for the current policy π, and all states s and action a. The transition from state-action pair to state-action pair must be considered and the value of the state-action pair must be

```
Initialize $V(s)$ arbitrary, $\pi$ the policy to be evaluated
Repeat (for each episode):
        Initialize $s$
        Repeat (for each step of episode):
            $a \leftarrow$ action given by $\pi$ for $s$
            Take action $a$: observe reward, $r$, and next state, $s^{'}$
            $V(s) \leftarrow V(s) + \alpha[r + \gamma V(s^{'}) - V(s)]$
            $s \leftarrow s^{'}$
        until $s$ is terminal
```

Figure A.6: Tabular TD(0) for estimating V^{π} [79].

learned. The theorems of convergence of state values under TD(0) also apply here for the action value algorithms. The equation for updating Q(s_t, a_t)) can be represented as follow [79]:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)]. \tag{A.10}$$

If s_{t+1} is terminal then $Q(s_{t+1}, a_{t+1})$ is zero. The general algorithm for the Sarsa method is presented in Figure A.7. The policy's dependence on Q defines the convergence properties of Sarsa.

```
Initialize $Q(s,a)$ arbitrary
Repeat (for each episode):
        Initialize $s$
        Choose $a$ from $s$ using policy derived from $Q$ (e.g., $\epsilon$-greedy)
        repeat (for each step of episode):
            Take action $a$, observe $r, s^{'}$
            Choose $a^{'}$ from $s^{'}$ using policy derived from $Q$ (e.g., $\epsilon$-greedy)
            $Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma Q(s^{'}, a^{'}) - Q(s,a)]$
            $s \leftarrow s^{'}, a \leftarrow a^{'}$
until $s$ is terminal
```

Figure A.7: Sarsa: An on-policy TD control algorithm [77].

Appendix B

Opposition-Based Learning

In this appendix, the outlines of Opposition-Based Learning (OBL), a new scheme for machine intelligence, is introduced [85]. Diverse forms of the concept of opposition are already existent in philosophy, linguistics, psychology and physics. The interplay between entities and their opposites is apparently fundamental for the maintenance of balance in a universal manner. However, it seems that we have failed to incorporate oppositional thinking in engineering, mathematics and computer science. Considering estimates along with counter-estimates can establish a new learning scheme with a wide range of applications. For instance, hyper-dimensionality and convergence speed are closely related and major challenges in machine learning. A large number of problems in engineering and science cannot be approached with conventional schemes and are generally handled with intelligent techniques such as evolutionary, neural, reinforcing and swarm-based techniques. These methodologies, however, suffer from long training or optimization time. The underlying idea of OBL is simultaneous consideration of guess and opposite guess, estimate and opposite estimate, in order to accelerate learning, search and optimization. In this appendix definitions will be established.

B.1 What is Opposition?

Opposition is related to entities, objects or their abstractions of the same nature that are completely different in some manner. For instance, cold and hot both describe a certain temperature perception of the same kind. Both however, are completely different since they are located at opposite spots of the temperature scale. Transition from one entity to its opposite can naturally establish rapid and fundamental changes. Social revolutions, for instance, aim mainly at attaining opposite circumstances, dictatorship versus democracy, by initiating sudden transitions. This may be taken as an initial thought for our purpose of namely accelerating algorithms by smart selection between estimate and opposite estimate. In natural language, opposition can be detected at different levels: Directional opposition (north-south, up-down, left-right), adjectival opposition (ugly-handsome, long-short, high-low), and prefix opposition (thesis vs. anti-thesis, revolution vs. counter-revolution, direction vs. opposite direction). Further, one can distinguish complements (mutually exclusive properties: dead-alive, true-false), antonyms (two corresponding points or ranges of a scale: long-short, hot-cold), directional converses (two directions along an axis: east-west, up-down), and relational converses (the relative positions of two entities on opposite sides: above-below, teacher-pupil). Human communication without utilizing linguistic opposition is unimaginable. Many examples also exist in philosophies which deal with opposition. Rewards versus punishments, and positive versus negative reinforcements have been intensively investigated. Every reinforcement signal has its own distinct influence, which may not be easily achieved with the opposite reinforcement signal. These examples show that opposition plays a central role in many fields. In spite of all these examples, it should be mentioned that understanding and defining opposition may not be straightforward in some cases.

Learning, optimization and search are fundamental tasks in machine intelligence research. Algorithms learn from the past data or instructions, optimize estimated solutions and search in large spaces for an existing solution. The problems can be of a different nature, and algorithms are inspired by diverse biological, behavioural and natural phenomena. Whenever we are looking for the solution u of a given problem, we usually make an estimate $\hat{u}$. This estimate is not the exact solution and could be based on experience or a totally random guess. In some cases we are satisfied with the estimate $\hat{u}$ and sometimes we try further to increase the resulting accuracy, if possible. In this sense, if we understand the task of many intelligent techniques as function approximation, then we generally have to cope with computational complexity. In many cases the learning begins at a random point. We begin from scratch and move toward an existing solution.

In a reinforcement learning agents, the action policy is initially based on randomness. The random guess, if not far away from the optimal solution, can result in a fast convergence. However, it is natural to state that if we begin with a random guess, which is very far away from the existing solution, let us say that in the worst case it is in the opposite location. The approximation, search or optimization will then take considerably more time, or in the worst case become intractable. Of course, in absence of any a priori knowledge, it is not possible to make the best initial guess. Logically, we should be looking in all directions simultaneously, or more concretely, in the opposite direction. Searching in opposite direction could be beneficial from an algorithmic point of view as well. If we are searching for the solution u, and if we agree that searching in opposite direction could be advantageous in some cases, then calculating the opposite number $\breve{u}$ is the first step [84, 85].

Definition (Type-I Opposition)

- Let $P_u = (u_1, u_2, \cdots, u_n)$ be a point in an n-dimensional space, where $u_i \in [a_i, b_i]$ with $a_i, b_i \in R$. The type-I opposite point $\breve{P}_u = (\breve{u}_1, \breve{u}_2, \cdots, \breve{u}_n)$ is then completely defined where

$$\breve{u}_i = a_i + b_i - u_i. \tag{B.1}$$

Figure B.1 demonstrates a simple definition of the idea for one-dimensional equation based on the distance of the opposite guess from the interval boundaries.

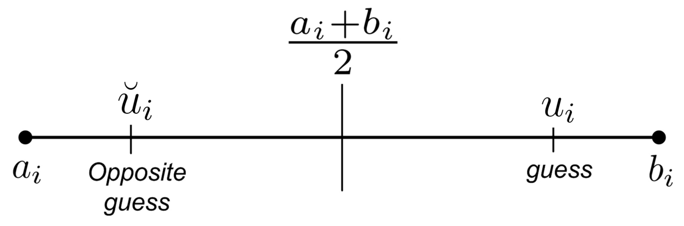

Figure B.1: A demonstration for one-dimensional case based on the distance of the opposite guess from the interval boundaries.

B.2 OBL Algorithms

In this section we establish a general framework for explicit employment of OBL within existing methodologies. For this purpose we begin a general definition of OBL algorithms.

Definition (OBL Algorithms)

- Any algorithm dealing with an unknown function $Y = f(X)$ and a known evaluation function $g(X)$ extended with OBL to calculate opposite $\breve{X}$ of numbers, guesses or estimates X for considering max $(g(X), g(\breve{X}))$ in its decision-making is an OBL algorithm. Generally, three classes of OBL algorithms can be distinguished:

1. Initializing OBL Algorithms

The concept of opposition is only used during the initialization. The effect is a better start with initial estimates closer to solution vicinity. Further, since it is done before the actual learning/searching begins, this creates no additional overhead.

2. Modifying OBL Algorithms

This approach is based on modification of the body of an existing algorithm. The effect of OBL will be much more visible, since opposition is considered in every iteration/episode/generation.

3. Initializing and Modifying OBL Algorithms

This class uses OBL during both initialization and actual learning by combining the two previous approaches.

The selection of an appropriate OBL algorithm directly depends on the problem at hand. However, the highest level of convergence speed can generally be achieved for the third class of OBL algorithms. The second and third class of OBL algorithms need to define the OBL condition, the condition that determines when, within the learning/search loop of the methodology, the OBL part should be activated.

Appendix C

Extraction of a Particular Object

C.1 RIS Configuration

A case where the application of the local RIS system is utilized, is the extraction of a particular object from an image. We assume that a threshold value exists, that can reveal the object fragments located in each sub-image. Due to possible non-uniform illumination, there could exist some artifacts as attached pieces to the object after thresholding. A morphological closing can be used as post-processing to remove them. The RIS controls the threshold value and the size of the morphological opening.

We implement the RIS system to images containing the object shown in figure C.1.

The structure of the RIS system for this problem is the same as the general framework in Chapter 4, with the following characteristics:

- In terms of algorithms, there are two processing tasks, thresholding and morphological opening, applied to each sub-image. The RL agent adjusts the parameters of these tasks locally.

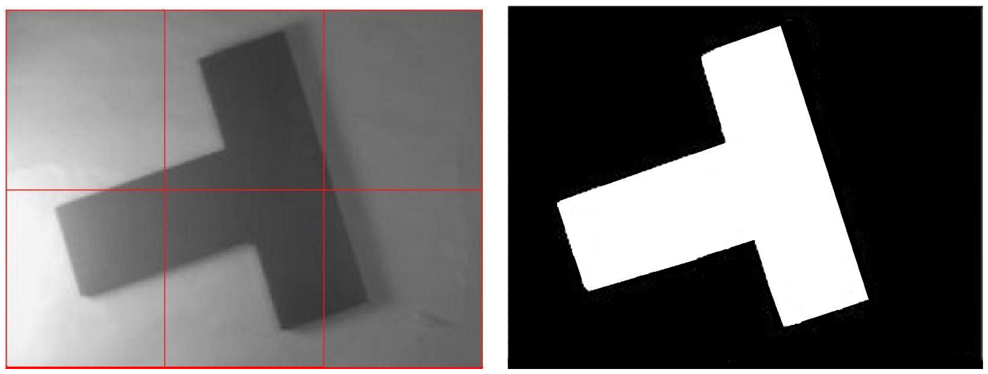

Figure C.1: The object to be extracted locally in RIS system (used for offline learning).

- For learning, the knowledge acquired during the offline mode is used on the real-time cases, while objective online learning is applied as well.

States, actions and reward are defined as follows:

States

The following features used in the Section 5.2.1(page 79) are chosen in each sub-image to define the state vector:

Feature 1: *Area* (A_{norm})

Feature 2: *Compactness* (Ψ)

Feature 3: *Gray Level Information*

Feature 4: *Number of Objects*(N_O)

Actions

The threshold value and the size of the structuring element are adjusted by RL actions. For each sub-image, the assigned local threshold value τ_i, spaced between the local maximum gray level g_{lmax} and local minimum gray level g_{lmin}, is changed by $\Delta_g = \frac{g_{lmax} - g_{lmin}}{10}$. For the morphological opening, the size of the structuring element is changed by 1 in the set $\{0, ..., 5\}$.

Defining Reward/Punishment

For offline reward/punishment, we measure the similarity between the ideal and extracted object as described in Section 5.2.1 (page 82).

C.2 Offline RIS and Testing

The offline training mode is performed using the object in Figure C.1. The image is divided in 6 sub-images and the RL agent works on each of them. After training, the RIS can be used on new images containing the same object under different circumstances (illumination, position, rotation etc.). The agent must find the appropriate local parameters and extract the object. To evaluate the results for the new images, we follow the same manner of online objective evaluation as presented in Section 5.2.1 (page 83).

Nine different images containing the same object were used to test the system. Figure C.2 shows 5 sample images and the results of the RIS system. The comparison (shown in the same Figure) has been made using Kittler and Otsu methods for each sub-image followed by a morphological opening. In all cases, Otsu method performs better than Kittler. Using the available central point (given by the user), only the object of interest, including any connected pieces, is selected in the output image of Otsu method. It is to create even better results for this method.

The results show that the proposed system is superior to the Otsu method. Table C.1 shows the area overlap with respect to the manually segmented image for all 10 images.

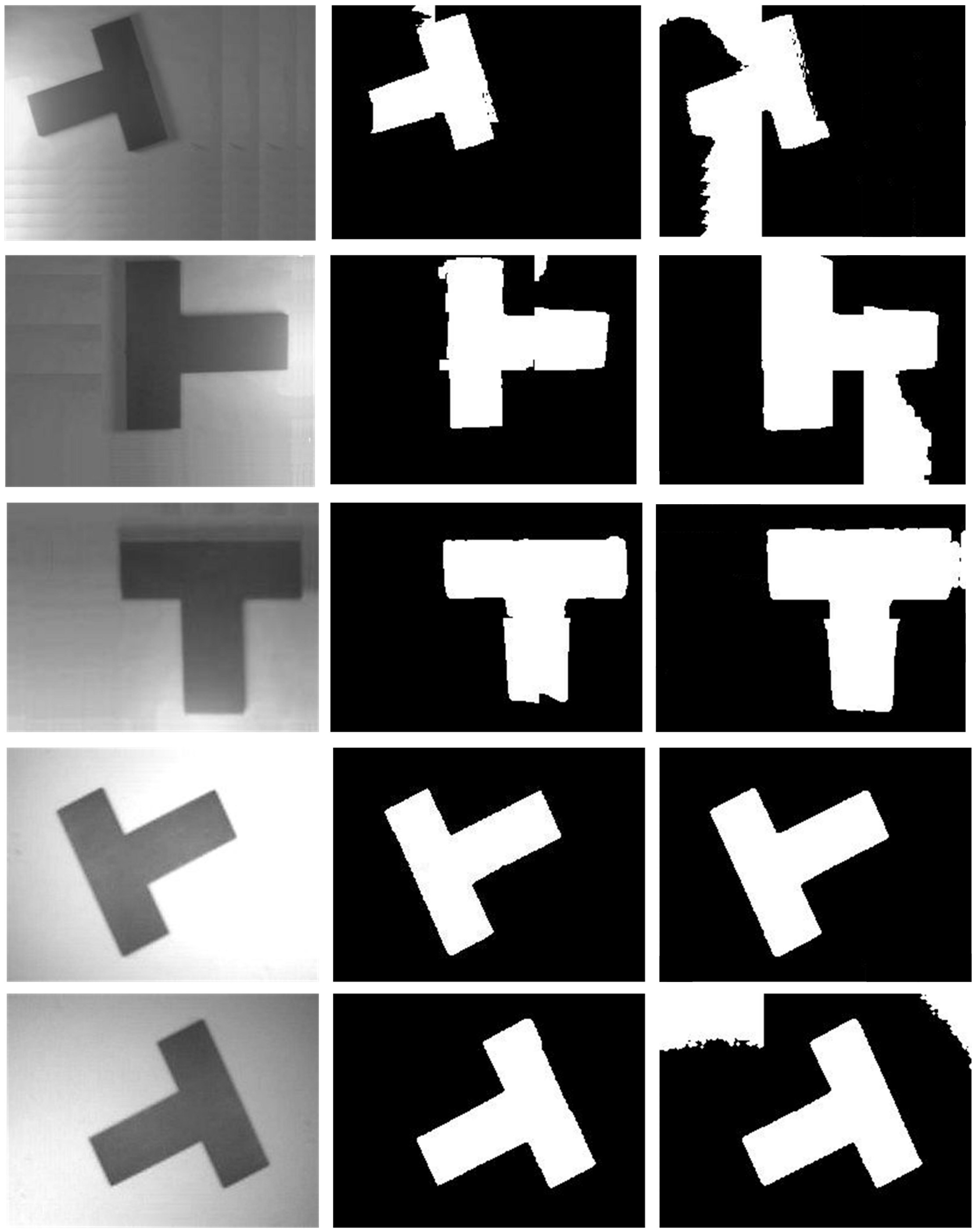

Figure C.2: Five sample images containing the object of interest. First, second and third columns show the original image, results of RIS system and results of Otsu method, respectively.

Table C.1: Area Overlap *AO* for training and testing images of the object in Figure C.1. Image number 1 is used in offline mode and the others for testing the system

Image	1	2	3	4	5	6	7	8	9	10
AO(%)	98.71	95.84	95.59	96.21	94.58	93.17	96.02	93.53	95.85	96.04

$\mu = 95.56$	$\sigma = 1.5$

Bibliography

[1] Cancer facts and figures. *American Cancer Society http://www.cancer.org.*

[2] G. B. Aboutanos and B. M. Dawant. Automatic brain segmentation and validation: image-based versus atlas-based deformable models. *SPIE Medical Imaging*, 3034:299–310, 1997.

[3] R. Adams and L. Bischof. Seeded region growing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16:641–647, 1994.

[4] M. J. Aitkenhead and A. J. S. McDonald. A neural network based obstacle-navigation animat in a virtual environment. *Engineering Applications of Artificial Intelligence*, 15:229–239, 2002.

[5] M. R. Anderberg. *Cluster Analysis for Applications.* Academic Press, New York, NY, 1973.

[6] M. Asada, S. Noda, S. Tawaratsumida, and K. Hosoda. Purposive behavior acquisition for a real robot by vision-based reinforcement learning. *Machine Learning*, 23:279–303, 1996.

[7] E. Bardinet, L. D. Cohen, and N. Ayache. A parametric deformable model to fit unstructured 3d data. *Computer Vision and Image Understanding*, 71:39–54, 1998.

[8] N. Betrounia, M. Vermandela, D. Pasquierc, S. Maoucheb, and J. Rousseaua. Segmentation of abdominal ultrasound images of the prostate using a priori information and an adapted noise filter. *Computerized Medical Imaging and Graphics*, 29:43–51, 2005.

[9] J. C. Bezdek, L. O. Hall, and L. P. Clarke. Review of mr image segmentation techniques using pattern recognition. *Medical Physics*, 20:1033–1048, 1993.

[10] B. Bhanu and J. Peng. Adaptive integrated image segmentation and object recognition. *IEEE Transactions on Systems, Man, and Cybernetics*, 30:427–441, 2000.

[11] S. T. Bow. *Pattern Recognition and Image Preprocessing.* Marcel Dekker Inc., New York, NY, 1992.

[12] A. Cavallaro, E. Drelie, and T. Ebrahimi. Objective evaluation of segmentation quality using spatio-temporal context. In *IEEE International Conference of Image Processing*, pages 301– 304, 2002.

[13] H. Cheng, Y. Lui, and R. Freimanis. novel approach to microcalcification detection using fuzzy logic technique. *IEEE Transactions on Medical Imaging*, 17:442–450, 1998.

[14] B. Chiu, G. H. Freeman, M. M. A. Salama, and A. Fenster. Prostate segmentation algorithm using dyadic wavelet transform and discrete dynamic contour. *Physics in Medicine and Biology*, 49:4943–4960, 2004.

[15] L. Cohen. Note on active contour models and balloons. *Image Understanding*, 53:211–218, 1991.

[16] D. L. Collins, C. Holmes, T. M. Peters, and A. Evans. Automatic 3-d model-based neuroanatomical segmentation. *Human Brain Mapping*, 3:190–208, 1995.

[17] P. L. Correia and F. Pereira. Objective evaluation of relative segmentation quality. In *IEEE International Conference on Image Processing*, pages 308–311, 2000.

[18] P. L. Correia and F. Pereira. Stand-alone objective segmentation quality evaluation. *EURASIP Journal on Applied Signal Processing*, pages 389–400, 2002.

[19] T. R. Crimmins. A complete set of fourier descriptors for two dimensional shapes. *IEEE Transactions System, Man and Cybernetics*, 12:848–855, 1982.

[20] C. Davatzikos and R. Bryan. Using a deformable surface model to obtain a shape representation of the cortex. *IEEE Transation on Medical Imaging*, 15:785–795, 1996.

[21] M. P. Eckert and A. P. Bradley. Perceptual quality metrics applied to still image compression. *Signal Processing*, 70:177–200, 1998.

[22] S. Geman and D. Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:721–741, 1984.

[23] P. Gibbs, D. Buckley, and A. Horsman S. Blackband. Tumor volume detection from mr images by morphological segmentation. *Physics in Medicin Biology.*, 41:2437–2446, 1996.

[24] L. Gong, S. Pathak, D. Haynor, P. Cho, and Y. Kim. Parametric shape modelling using deformable super ellipses for prostate segmentation. *IEEE Transactions on Medical Imaging*, 23:340–349, 2004.

[25] R. C. Gonzalez and R. E. Woods. *Digital Image Processing.* Addison-Wesley, second edition, 2002.

[26] L. O. Hall, A. Bensaid, L. P. Clarke, R. Velthuizen, M. S. Silbiger, and J. C. Bezdek. A comparison of neural network and fuzzy clustering techniques in segmenting magnetic resonance images of the brain. *IEEE Transactions on Neural Network*, 3:672–682, 1992.

[27] R. M. Haralick and L. G. Shapiro. Image segmentation techniques. *Computer Vision, Graphics, and Image Processing*, 29:100–132, 1985.

[28] K. Held, E. R. Kops, B. J. Krause, W. M. Wells, and R. Kikinis. Markov random field segmentation of brain mr images. *IEEE Transactions on Medical Imaging*, 16:878–886, 1997.

[29] M. Iida, M. Sugisaka, and K. Shibata. Application of direct-vision-based reinforcement learning to a real mobile robot. In *Proceedings of the International Conference of Neural Information Processing Systems*, 2002.

[30] M. F. Insana and D. G. Brown. *Acoustic scattering theory applied to soft biological tissues*. CRC Press, Boca Raton, 1993.

[31] R. Jain, R. Kasturi, and B. G. Schunck. *Machine Vision*. McGraw-Hill Inc., New York, NY, 1995.

[32] L. P. Kaelbling and M. L. Littman. Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, 4:237–285, 1995.

[33] T. Kapur, W. Grimson, R. Kikinis, and W. Wells. Enhanced spatial priors for segmentation of magnetic resonance imagery. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer-Verlag, 1998.

[34] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1:321–331, 1988.

[35] J. Kittler and J. Illingworth. On threshold selection using clustering criteria. *IEEE Transactions on Systems, Man, and Cybernetics*, 15:652–655, 1985.

[36] K. R. R. Krishnan, E. Gelenbe, and Y. Feng. Neural network methods for volumetric magnetic resonance imaging of the human brain. *Proceedings of the IEEE ISSN*, 84:1488–1496, 1996.

[37] H. M. Ladak, F. Mao, Y. Wang, D. B. Downey, D. A. Steinman, and A. Fenster. Prostate boundary segmentation from 2d ultrasound images. *Medical Physics*, 27:1777–1788, 2000.

[38] T. Lei and W. Sewchand. Statistical approach to x-ray ct imaging and its applications in image analysis. *IEEE Transactions on Medical Imaging*, 11:62–69, 1992.

[39] Z. Liang, J. MacFall, and D. Harrington. Parameter estimation and tissue segmentation from multispectral mr images. *IEEE Transactions on Medical Imaging*, 13:441–449, 1994.

[40] L. Lucchese and S. Mitra. Color image segmentation: A state-of-the-art survey. *Indian National Science Academy*, 67A:207–221, 2001.

[41] J. F. Mangin, V. Frouin, I. Bloch, J. Regis, and J. Lopez-Krahe. From 3d magnetic resonance images to structural representations of the cortex topography using topology preserving deformations. *Mathematical Imaging and Vision*, 5:297–318, 1995.

[42] I. N. Manousakas, P. E. Undrill, G. G. Cameron, and T. W. Redpath. Split-and-merge segmentation of magnetic resonance medical images: performance evaluation

and extension to three dimensions. *Computers and Biomedical Research*, 31:393–412, 1998.

[43] C. Mettlin. American society national cancer detection project. *Cancer*, 75:17901794, 1995.

[44] I. Mikic, S Krucinski, and J. D. Thomas. Segmentation and tracking in echocardiographic sequences: active contours guided by optical flow estimates. *IEEE Transation on Medical Imaging*, 17:274–284, 1998.

[45] A. Neumann and C. Lorenz. Statistical shape model based segmentation of medical images. *Computerized Medical Imaging and Graphics*, 22:133–143, 1998.

[46] J. A. Noble and D. Boukerroui. Ultrasound image segmentation: A survey. *IEEE Transactions on Medical Imaging*, 25:987–1010, 2006.

[47] N. Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9:62–66, 1979.

[48] M. Ouslim and K.M. Curtis. Machine parts classification based on a digital neural network. pages 643–646, 1996.

[49] N.R. Pal and S.K. Pal. A review on image segmentation techniques. *Pattern Recognition*, 26:1277–1294, 1993.

[50] T. N. Pappas. An adaptive clustering algorithm for image segmentation. *IEEE Transactions on Signal Processing*, 40:901–914, 1992.

[51] S. D. Pathak, D. R. Haynor, and Y. Kim. Edge-guided boundary delineation in prostate ultrasound images. *IEEE Transactions on Medical Imaging*, 19:1211–1219, 2000.

[52] J. Peng and B. Bhanu. Closed-loop object recognition using reinforcement learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20:139–154, 1998.

[53] D. Pham and J. L. Prince. An adaptive fuzzy c-means algorithm for image segmentation in the presence of intensity inhomogeneities. *Pattern Recognition Letter*, 20:57–68, 1999.

[54] S. Pohlman, K. Powell, N. Obuchowski, W. Chilcote, and S. Broniatowski. Quantitative classification of breast tumors in digitized mammograms. *Medical Physics*, 23:1337–1345, 1996.

[55] W. Polakowski, D. Cournoyer, S. Rogers, M. DeSimio, and D. Ruck. Computer-aided breast cancer detection and diagnosis of masses using difference of gaussians and derivative-based feature saliency. *Transactions on Medical Imaging*, 16:811–819, 1997.

[56] J. S. Prater and W. D. Richard. Segmenting ultrasound images of the prostate using neural networks. *Ultrasound Imaging*, 14:159–185, 1992.

[57] C. Riberio. Reinforcement learning agent. *Artificial Intelligence Review*, 17:223–250, 2002.

[58] W. Rodriguez, M. Lastb, A. Kandel, and H. Bunked. 3-dimensional curve similarity using string matching. *Robotics and Autonomous Systems*, 49:165–172, 2004.

[59] S. J. Russell and P. Norvig. *Artificial Intelligence : a Modern Approach.* Prentice Hall, Englewood Cliffs, NJ, 1995.

[60] F. Sahba and H. R. Tizhoosh. Filter fusion for image enhancement using reinforcement learning. In *Canadian IEEE Conference on Electrical and Computer Engineering, Montreal, Quebec, Canada*, pages 847–850, 2003.

[61] F. Sahba, H. R. Tizhoosh, and M. M. A. Salama. A new technique for adaptive fuzzy image enhancement for edge detection. In *International Workshop on Multidisciplinary Image, Video, and Audio Retrieval and Mining,Sherbrooke*, page 14, 2004.

[62] F. Sahba, H. R. Tizhoosh, and M. M. A. Salama. A coarse-to-fine approach to prostate boundary segmentation in ultrasound images. *BioMedical Engineering*, 4:58, 2005.

[63] F. Sahba, H. R. Tizhoosh, and M. M. A. Salama. Segmentation of prostate boundaries using regional contrast enhancement. In *The IEEE International Conference on Image Processing (ICIP), Genova, Italy*, pages 1266–1269, 2005.

[64] F. Sahba, H. R. Tizhoosh, and M. M. A. Salama. Using reinforcement learning for filter fusion in image enhancement. In *The Fourth IASTED International Conference on Computational Intelligence, Calgary*, pages 262–266, 2005.

[65] F. Sahba, H. R. Tizhoosh, and M. M. A. Salama. Increasing object recognition rate using reinforced segmentation. In *The IEEE International Conference on Image Processing (ICIP), Atlanta*, pages 781–784, 2006.

[66] F. Sahba, H. R. Tizhoosh, and M. M. A. Salama. A reinforcement learning framework for medical image segmentation. In *The IEEE World Congress on Computational Intelligence (WCCI), Vancouver*, pages 1238–1244, 2006.

[67] F. Sahba, H. R. Tizhoosh, and M. M. A. Salama. Application of opposition-based reinforcement learning in image segmentation. In *IEEE Symposium on Computational Intelligence in Image and Signal Processing (CIISP),Hawaii*, pages 246–251, 2007.

[68] P. Sahoo, S Soltani, A. Wong, and Y. Chen. Survey of thresholding techniques. *Computer Vision, Graphics and Image Processing*, 41:233–260, 1988.

[69] S. Sandor and R. Leahy. Surface-based labeling of cortical anatomy using a deformable atlas. *IEEE Transactions on Medical Imaging*, 16:41–54, 1997.

[70] B. Sankur and M. Sezgin. A survey over image thresholding techniques and quantitative performance evaluation. *Journal of Electronic Imaging*, 13:146–168, 2004.

[71] R. J. Schalkoff. *Pattern Recognition: Statistical, Structural and Neural Approaches.* Wiley and Sons, New York, NY, 1992.

[72] D. Shen, Y. Zhan, and C. Davatzikos. Segmentation of prostate boundaries from ultrasound images using statistical shape model. *IEEE Transactions on Medical Imaging*, 22:539–551, 2003.

[73] K. Shibata and M. Iida. Acquisition of box pushing by direct-vision-based reinforcement learning. In *Proceedings of the Society of Instrument and Control Engineers Annual Conference*, 2003.

[74] M. Shokri and H. R. Tizhoosh. Using reinforcement learning for image thresholding. In *Canadian IEEE Conference on Electrical and Computer Engineering*, 2003.

[75] M. Shokri and H. R. Tizhoosh. Q(λ)-based image thresholding. In *Canadian Conference on Computer and Robot Vision*, 2004.

[76] J. Sijbers, P. Scheunders, A. Van Der Linden M. Verhoye, and D. Van Dyck. Watershed-based segmentation of 3d mr data for volume quantization. *Magnetic Resonance Imaging*, 15:679–688, 1997.

[77] S. Singh, P. Norving, and D.Cohn. *Introduction to Reinforcement Learning.* Harlequin Inc., 1996.

[78] M. Sonka, V. Hlavac, and R. Boyle. *Image Processing, Analysis and Machine Vision.* International Thompson Computer Press, 1993.

[79] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction.* MIT Press, Cambridge, MA, 1998.

[80] J. A. Swets. *Signal Detection Theory and Roc Analysis in Psychology and Diagnostics.* Lawrence Erlbaum Associates, Mahwah, NJ, 1996.

[81] G. A. Taylor. A reinforcement learning framework for parameter control in computer vision applications. In *First Canadian Conference on Computer and Robot Vision*, 2006.

[82] S. Theodoridis and K. Koutroumbas. *Pattern Recognition; 3 edition.* Academic Press, San Diego, CA, 2006.

[83] H. R. Tizhoosh. *Fuzzy Image Processing.* Springer, 1997.

[84] H. R. Tizhoosh. Opposition-based learning: A new scheme for machine inteligence. In *International Conference on Computational Inteligence for Modeling Control and Automation,Vienna,Austria*, pages 695–701, 2005.

[85] H. R. Tizhoosh. Reinforcement learning based on actions and opposit actions. In *International Conference on Artificial Inteligence and Machine Learning,Cairo,Egypt*, 2005.

[86] H. R. Tizhoosh and G. W. Taylor. Reinforced contrast adaptation. *International Journal of Image and Graphic*, 6:377–392, 2006.

[87] J. K. Udupa and S. Samarasekera. Fuzzy connectedness and object definition: Theory, algorithms and applications in image segmentation. *Graphical Models and Image Processing*, 58:246–261, 1996.

[88] L. Vincent and P. Soille. Watersheds in digital spaces: an efficient algorithm based on immersion simulation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13:583–598, 1991.

[89] Y. Wang, H. Cardinal, D. Downey, and A. Fenster. Semiautomatic three-dimensional segmentation of the prostate using two dimensional ultrasound images. *Medical physics*, 30:887–897, 2003.

[90] C. J. C. H. Watkins. *Learning from Delayed Rewards.* Cambridge, Cambridge University, 1989.

[91] C. J. C. H. Watkins and P. Dayan. Q-learning. *Machine Learning*, 8:279–292, 1992.

[92] P. Wust, J. Gellermann, J. Beier, S. Wegner, and W. Tilly. Evaluation of segmentation algorithms for generation of patient models in radiofrequency hyperthermia. *Physics in Medicin and Biology*, 43:3295–3307, 1998.

[93] P. Y. Yin. Maximum entropy-based optimal threshold selection using deterministic reinforcement learning with controlled randomization. *Signal Processing*, 82:993–1006, 2002.

[94] P. Zamperoni. Analysis of some region growing operators for image segmentation. *Advances in Image Processing and Pattern Recognition*, pages 204–208, 1986.

[95] D. Zhang and G. Lu. Study and evaluation of different fourier methods for image retrieval. *Image and Vision Computing*, 23:33–49, 2005.

[96] Y. J. Zhang. Evaluation and comparison of different segmentation algorithm. *Pattern Recognition Letters*, 18:963–974, 1997.