# Selecting Automatically Vision Operators with their Parameters to Accomplish a Segmentation Task

Issam Qaffou
Département Informatique, FSSM
Université Cadi Ayyad,
Marrakech, Morocco

Mohamed Sadgal
Département Informatique, FSSM
Université Cadi Ayyad,
Marrakech, Morocco

Aziz Elfazziki
Département Informatique, FSSM
Université Cadi Ayyad,
Marrakech, Morocco

## ABSTRACT
In a vision system, every task needs that the operators to apply should be « well chosen » and their parameters should be also « well adjusted ». The diversity of operators and the multitude of their parameters constitute a big challenge for users. As it is very difficult to make the « right » choice, lack of a specific rule, many disadvantages appear and affect the computation time and especially the quality of results. In this paper we present a multi-agent architecture to learn the best operators to apply and their best parameters for a class of images. Our architecture consists of three types of agents: User Agent, Operator Agent and Parameter Agent. The User Agent determines the phases of treatment, a library of operators and the possible values of their parameters. The Operator Agent constructs all possible combinations of operators and the Parameter Agent, the core of the architecture, adjusts the parameters of each combination by treating a large number of images. Through the reinforcement learning mechanism, our architecture does not consider only the system opportunities but also the user preferences.

## Keywords
Computer Vision, Reinforcement Learning, Multi-Agent System, Parameter Adjustment, Operator Selection, Q-learning, Segmentation.

## 1. INTRODUCTION
To accomplish an image processing task (segmentation, detection, object recognition, etc.) the user finds him-self faced with a multitude of applicable operators averaging the fixation of values for several parameters. The quality of results depends essentially on the operator chosen and the values assigned to its parameters. The lack of a general rule that guides the user in his choices pushes him usually to use his experience and sometimes his intuition. He, generally, proceeds by trial and error until the identification of a satisfactory result. The problem is already remarkable when the task needs to apply just one operator but with several parameters to adjust. However, in the majority of vision tasks, the user is required and sometimes even is obliged to combine several operators whose each one has a multitude of parameters to adjust. Therefore the user must select the operators, adjust their parameters and then test them sequentially on the image. This process is repeated for a long time before deciding on the quality of the results. It's a tedious work with a great waste of time. To accomplish his application, the user reuses usually the last combination of operators that he has found. But, it is possible that there is a better combination that has not been tested by the user. The exploration and the exploitation of all possible combinations constitute a source of errors before talking about the time spent in the operation. To help the user to perform vision tasks, several solutions have been proposed as systems and GUI. For example, Pandore and Ariane [1]. These semi automatic solutions provide a library of operators and a set of parameters for each operator, the selection of operators and the adjustment of their parameters are done manually by the user by using a GUI. Even though, the user finds always difficulty to choose the appropriate operators and adjust their parameters in order to find the best result.

Some authors searched to automate the operator selection process. Draper proposed ADORE in 2000. It is a system of object recognition based on MDP (Markov Decision Process) to choose, from a current situation, the operator to apply [2]. Draper used a library of ten operators to recognize duplexes in aerial images. ADORE is based on a method which is robust theoretically, but which cannot always ensure good results because, on one hand, Draper uses a predefined and limited library of operators, and on the other hand he didn't talk about the problem of parameter adjustment. Other authors proposed methods to automatically adjust parameters of vision operators. B.NICKOLAY et al. proposed a method to automatically optimize the parameters of a machine vision system for surface inspection by using specific Evolutionary Algorithms (EA) [3]. A few years later, Taylor [4] proposed a reinforcement learning framework which uses connectionist systems as function approximators to handle the problem of determining the optimal parameters for a computer vision application even in the case of a highly dimensional, continuous parameter space. More recently, Farhang et al. [5] introduced a new method for segmentation of the prostate in transrectal ultrasound images, using a reinforcement learning (RL) scheme. He divided the initial image into sub-images and works on each sub-image in order to reach a good result. In [6] and [7], we proposed a reinforcement learning method to adjust automatically the parameters of vision operators. Our method is general for every vision task using parametric operators. The programmer has only to determine the combination of operators to apply and their parameters to adjust and it is to our method to find automatically the best values of these parameters depending on the task at hand. Despite all these researches and their results, they stay limited to a predefined type of images or depend on some particular conditions. That's on one hand; on the other hand they don't propose general solution for the problem of operator selection and parameters adjustment. Thus, until today, there is no method robust, sure and automatic which provides the user the appropriate operators and their optimal parameters values depending on the vision task and the class of images. Hence, we need systems that allow, generally and for any vision task, to automatically determine the best combination of operators and their optimal parameters values to apply.

In this paper we present a solution for this problem by proposing a multi-agents architecture based on reinforcement learning to select automatically the best operators to apply in a vision task (segmentation, extraction, recognition, etc), that's while adjusting their parameters values without the user intervention. Our system consists of three types of intelligent agents: a User Agent (UA) charged to give all the necessary information to accomplish the vision task, an Operator Agent

(OA) charged to choose automatically the operators and a Parameter Agent (PA), the core of our architecture, charged to adjust parameters of each operator. The agents OA and PA communicate with each other to find the best operators and their best parameters values. More details about the task of each agent are explained below. In the second section we present an overview on reinforcement learning, multi-agent systems. The third section details the proposed approach. The forth section discusses the experience and its results. The last section concludes the paper.

# 2. OVERVIEW

In this section we present the two principle concepts which underlie our approach. In the first subsection, we discuss the reinforcement learning concept and its application in image processing. The second subsection concerns multi-agent systems and their use to accomplish vision tasks.

## 2.1 Reinforcement Learning

According to the definition of S.Sutto and G.Barto [8], reinforcement learning defines a type of interaction between an agent and its environment. From a real situation « s » in the environment, the agent chooses and executes an action « a » which causes a transition to the stat « s' ». It receives in return a reinforcement signal « r », which is a penalty if the action leads to a failure or a reward if the action is beneficial; a zero signal means the inability to assign a penalty or a reward. The agent uses then this signal to improve its strategy, action sequence, in order to maximize the accumulation of its future rewards. For this purpose, it must balance exploration and exploitation. The exploration is to test new action, which could lead to higher earnings. Whereas the exploitation consists to apply the best strategy previously acquired. Watkins has developed Q-learning, a well-established on-line learning algorithm, as a practical RL method [9]. In this algorithm, the agent maintains a numerical value for each state-action, representing a prediction of the worthiness of taking an action in a state. Table 1 represents an iterative policy evaluation for updating the state-action values where r is the reward value received for taking action a in state s, s' is the next state, α is the learning rate, and γ is the discount factor. There are some policies for taking action a given state s. One of them is the Boltzman policy which estimates the probability of taking each action in each state. There are other policies for Q-learning such as ε-greedy and greedy. In the greedy policy, all actions may not be explored, whereas the ε-greedy selects the action with the highest Q-value in the given state with a probability of 1 – ε and others with a probability of ε. In this work an ε-greedy policy is used to make a balance between exploration and exploitation. The reward r is defined according to each state-action pair (s, a). The goal is to find a policy to maximize the discounted sum of rewards received over time. The principal concerns in RL are the cases where the optimal solutions cannot be found, but can be approximated. Ideally, the RL agent does not require a set of training samples. Instead, it can continuously learn and adapt while performing the required task.

Table 1: Q-learning algorithm

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
  Take action $a$ , observe r, s'

$$Q^{\pi}(s_t,a_t) = (1-\alpha_t)Q^{\pi}(s,a) + \alpha_t(r + \gamma \max_{a \in A} Q_t^{\pi}(s',a'))$$

s←s';
Until s is terminal

## 2.2 Multi-agents systems

The agent concept has been studied for a long time in various disciplines. Multiple definitions of agent have been given depending on the field of application. In our work, we use the definition adopted by Haroun [10] based on M.Wooldridge's works: "an agent is a computer system, situated in some environment, that acts autonomously and flexibly in order to achieve its delegated goals".

A multi-agents system consists of a set of multiple agents living at the same time, sharing common resources and communicating with each other. The key point of multi-agents systems is the formalization of coordination between agents. The agents are able to perceive and act on a common environment that they share. Perceptions allow agents to acquire information about their environment evolution, and their actions allow them to change it.

# 3. PROPOSED APPROACH

Generally, to accomplish a vision task we've to pass through phases of processing. Each phase contains a set of operators, usually predefined in a system with their parameters whose some of their values are given by default. The users find themselves faced with a tedious work of choosing the best operator to apply and adjusting its parameters. In our approach we propose a multi-agents architecture which helps automatically the user in his choices (operators and parameters values). The architecture is composed globally by three types of agents. Each one of them is charged to accomplish one task in the process. Fig 1 shows how these agents are linked.

## 3.1 User Agent (UA)

Depending on the vision task to accomplish, UA gives the list of processing phases. For each phase, it determines a set of possible operators. For each operator it defines parameters to adjust by specifying ranges of their possible values. It also proposes a class of images for learning, on which the system will run, as well as a ground truth for each image. The work of UA is necessary so that the operator agent and the parameter agent can proceed.

## 3.2 Operator Agent (OA)

Operator agent proceeds in two steps: the first one is to build, according to the phases determined by UA, all possible combinations of operators. Each combination contains a number of operators which is equal to the number of the phases determined by UA. For each combination, the agent OA generates an agent PA (Parameter Agent) specialized to adjust parameters of its operators. There are then so many agents PA as possible combinations. Each agent PA has its own combination of operators. After adjusting parameters, according to the task at hand, each agent PA returns its combination of operators with the best parameters values. It also returns the result quality of this combination after applying it on the class of images determined by UA. The second step of the agent OA is to decide among all these combinations of operators which one

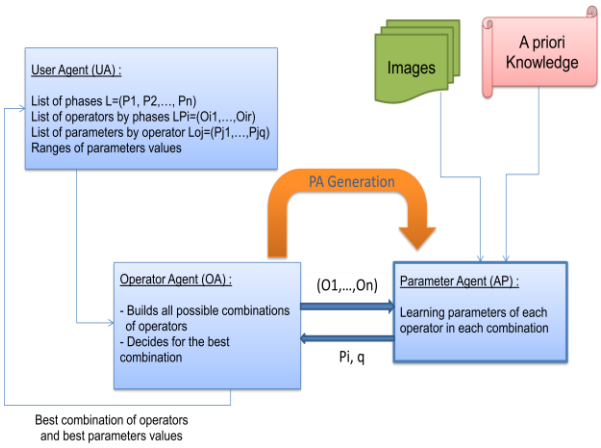


**Fig 1: global schema of the proposed approach. OA proceeds in collaboration with PA**

is the best to apply. The best combination corresponds to this one having the higher result quality; it is then returned to the agent UA.

Each agent PA uses reinforcement learning to adjust the parameters of each operator. It applies actions on a set of images and receives a return which may be a punishment or a reward. This return is determined depending to a ground truth proposed by an expert (manual processing). More details about how the agents PA proceed are given hereafter.

## 3.3 Parameter Agent (PA)

For each combination of operators, there is an agent PA to adjust parameters of these operators. To do this, a range of values for each parameter and a set of images with their ground truth are given by the agent UA. The agent PA has no prior knowledge about the best parameters values. It proceeds by reinforcement learning to find values giving the best result. Fig 2 presents a general schema about the functioning of each agent PA.

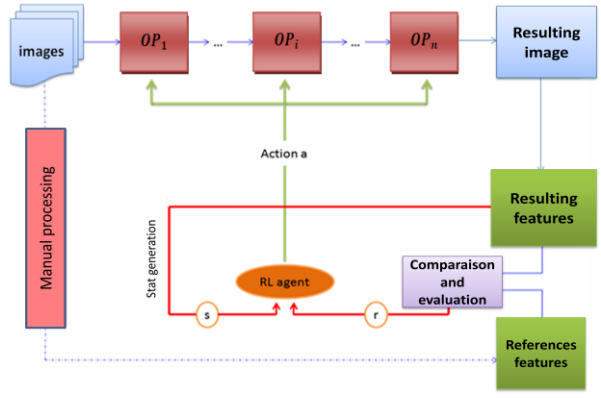


**Fig 2: general schema about the functioning of each agent PA for operator's parameter learning**

The input image is a processing subject of a series of operators. Each operator has a set of parameters to adjust, and each parameter has a range of possible values. The agent PA must find the best parameter values for each operator in order to get the best result. The agent PA uses reinforcement learning as an automatic method to explore all possible values and then exploit the best ones. The agent PA must then define the actions a, states s and reward r. We define actions as all possible combinations of parameters values. States are defined by features describing the image. These features are defined according to the task at hand. The agent PA chooses an action and applies the combination of operators on the input image and gets a resulting image. Each image has its ground-truth; it is a resulting image through a manual processing by an expert. The ground-truth represents a reference for the agent PA. To assess the chosen action, the agent PA compares the resulting image with the reference one. It extracts some features from the resulting image and compares them with the same features extracted from the reference image. An evaluation metric is used to assess the result and produce a reward. Each action has its own reward. The best action is the most rewarded. The details about how the three components, namely state, action, and reward are defined in our proposed approach are described in the next subsections.

*3.3.1 Defining actions:* Generally, all possible combination of parameters values of operators is defined as an action for the agent PA. The set of the actions is then the set of all possible values combination, see fig 2.

Each operator $OP_k$ has a series of parameters:

$$(P_1^k, P_2^k, ..., P_n^k)$$

Each parameter $P_j^k$ has a range of values:

$$V_j^k = \{V_{j1}^k, V_{j2}^k, ..., V_{jm}^k\}$$

An elementary action of the operator $OP_k$ is:

$$a_k = (u_{j1}^k, ..., u_{jr}^k) \text{ where } u_{j1}^k \in V_j^k$$

An action of the agent PA is defined by the combinations of the elementary actions of operators as it is defined above:

$$a = (a_1, a_2, ..., a_n)$$

*3.3.2 Defining states:* A state is defined by a set of features extracted from the resulting image:

$$s = [\chi_1, \chi_2, ..., \chi_n]$$

$\chi_i$ is a feature reflecting the state of the image after the processing. The type of the extracted features depends on the task at hand. Here we give a general definition, and in the experience we define them explicitly according to the application.

*3.3.3 Defining the reward:* The return is a reward if the agent PA chooses the right action, else it is a punishment. It is defined according to the quality of the processing result. This quality is assessed by using ground-truth models (manually processed images). To define the return we calculate the similarity between the resulting image and the ground truth image. That is depending on the task at hand. For example, if we use an edge detection approach for image segmentation we would calculate error measures which give global indices about the result quality: over-detection error, under-detection error, localization error [11]. But if we use a region approach we would calculate, for example, errors of Yasnoff [12] or the criterion of Vinet [12], etc. After measuring the similarity's criterions, we assess the result of our system using a weighted sum of the differences of these criterions' scalars:

$$D = \sum_i w_i D_i$$

The weights $w_i$ are chosen according to the importance of each criterion $D_i$.

In our experiments, we've used three error measures: over-detection error, under-detection error and localization error [11] which are formally expressed in the next section.

A general form of the reward definition in the proposed approach is presented by:

```
Reward: r= -10, 0 or 10;
    if (D < ε ) r = +10; f=true;
            else
            if ( (D > ε ) && (D < ε + δ) )
                r = 0;
                else r = -10;
            end
        end
```

The values 10 and -10 represent respectively the reward and the punishment depending to a predefined threshold.
Using the set of images determined by the agent UA, each agent PA returns to the agent OA its combination of operators with the best values of their parameters. It returns also the quality of the result corresponding to the highest reward. The agent OA retrieves then all the combinations it has built with the best parameters values of each operator and the qualities of their results. The agent OA returns to the agent UA the best combination of operators corresponding to the highest quality. Thus it decides which the best combination of operators to apply is.
In the following section we test this approach for segmentation tasks.

# 4. RESULTS AND DISCUSSION

In this section we test practically the multi agent architecture to choose the right operators and their best parameters values for segmentation tasks. The operators used in image segmentation differ from a class of images to another, they are not necessary the same. Our main goal in this section is to show how the proposed approach can determine, for a class of images, the best combination of operators to apply for their segmentation. The images used in the experience are those of traffic signs. The best operators are determined among those proposed by the agent UA. We use Matlab for the implementation.

## 4.1 The agent UA

It defines three phases of processing: preprocessing, processing and post processing. It defines also the operators for each phase and the possible values of their parameters.

*4.1.1 Preprocessing phase:* This phase consists to improve the quality of the image using filters. For this purpose, the agent UA proposes three operators. These operators are predefined in Matlab by: **'medfilt2'; 'ordfilt2'; 'wiener2 '.** The agent UA proposes just one parameter to adjust for all these operators: the size of the used filter. We define an operator by its name, the number of parameters and the list of their possible values.

**Op= {operator name, number of parameters, List of possible Values}**

The operators of the preprocessing phase are then defined as:
Op1 = {{'medfilt2'} {1} {[3 5]}};
Op2 = {{'wiener2'} {1} {[3 5]}};
Op3 = {{'ordfilt2'} {1} {[3 5]}};

*4.1.2 Processing phase*: This phase consists of detecting edges in the image. The agent UA proposes **'edge'**, a predefined operator in Matlab, as one operator for this phase with two parameters to adjust: the filter to select and the threshold to remove edges with poor contrast. Contours are formed by pixels higher than a given threshold. The operator **'edge'** is then defined as:

Op= {{'edge'} {2} {{'sobel' 'prewitt' 'zerocross' 'log'}; [0.02 0.03 0.04 0.05 0.06 0.07 0.08 0.09 0.1]}};

*4.1.3 Post processing phase:* This phase consists of refining the image by deleting small objects. The agent UA proposes **'bwareaopen'**, a predefined operator in Matlab, as one operator for this phase with two parameters to adjust: the connectivity and the maximal size of the objects to remove. The connectivity is defined by the number of neighbors to consider 4 or 8. For 2D images, the default connectivity is 8. **'bwareaopen'** is then defined as:

Op = {{'bwareaopen'} {2} {[5 10 15 20]; [8]}};

## 4.2 The agent OA

After receiving all the necessary information from the agent UA, the agent OA constructs all possible combinations of operators. As the agent UA determines three phases to accomplish the segmentation task, each one of these combinations will contain three operators.
The constructed combinations are then:
C1= (medfilt2, edge, bwareaopen)
C2= (wiener2, edge, bwareaopen)
C3= (ordfilt2, edge, bwareaopen)
For each combination, the agent OA generates an agent PA to adjust parameters according to the dataset of images proposed by the agent UA. There are then three agents PA1, PA2 and PA3 which treat respectively the combinations: C1, C2 and C3.

## 4.3 The agent PA

The agent PA is, generally, charged to adjust parameters of each operator in order that the segmentation result will be as close as possible to the segmentation done manually by an expert. To adjust parameters of each operator, the agent PA uses reinforcement learning. It must then define actions, states and reward. See the subsection III.C.

*Actions:* Actions are all possible combinations of parameters values. We select another action by choosing other parameters values. An example of an action for the agent PA1: Action= [3, ('sobel', 0.02), (5,8)]

*States:* States are defined by some features extracted from the image. In this application, we define a state by three features:

$$S = [\chi_1, \chi_2, \chi_3]$$

$\chi_1$ is the ratio between the number of contours in the resulting image and the number of those in the reference image (ground truth).

$\chi_2$ is the ratio between the total of white pixels in the resulting image and those in the reference image.

$\chi_3$ is the ratio between the length of the longest contour and the length of the longest contour in the reference image.

*Reward:* Reward is defined by a weighted sum of three error measures which give some global indices about the quality of boundary-based segmentation: over detection error, under-detection error and localization error [11]. These criteria evaluate a result of edge detection. The weights used in the definition of the reward are chosen according to the importance of each criterion. The reward is defined as:

$$D = w_1 D_1 + w_2 D_2 + w_3 D_3$$

where $w_i$ is a weight for $D_i$

and

$$D_1 = ERR_{sur}(I_r, I_{ref}) = \frac{card(I_r) - card(I_r \cap I_{ref})}{card(I_r) - card(I_{ref})}$$

and

$$D_2 = ERR_{sous}(I_r, I_{ref}) = \frac{card(I^{cont}_{ref/r})}{card(I^{cont}_{ref})}$$

and

$$D_3 = ERR_{loc}(I_r, I_{ref}) = \frac{card(I_{ref/r} \bigcup I_{r/ref})}{card(I_r)}$$

More details about $D_1$, $D_2$ and $D_3$ are given in [11].

For each combination of operators, the agent PA finds the best parameters values which give the best segmentation. In this experience, we test the proposed architecture to segment two different types of images. The first one is a dataset of 70 images of traffic signs and the second contains 60 real and highly textured images. Each image has its ground-truth. Fig 3 shows the result of segmentation for 5 images taken randomly from the processing of the dataset. It is important to note that we evaluate an operator on the whole of the dataset of images and not one by one. The fixation of some parameters of the Q-learning algorithm affects largely the result. The results showed in fig 4 are for: α=0.5, γ=0.8, ε=0.5, number of episodes=200 and number of steps=80.

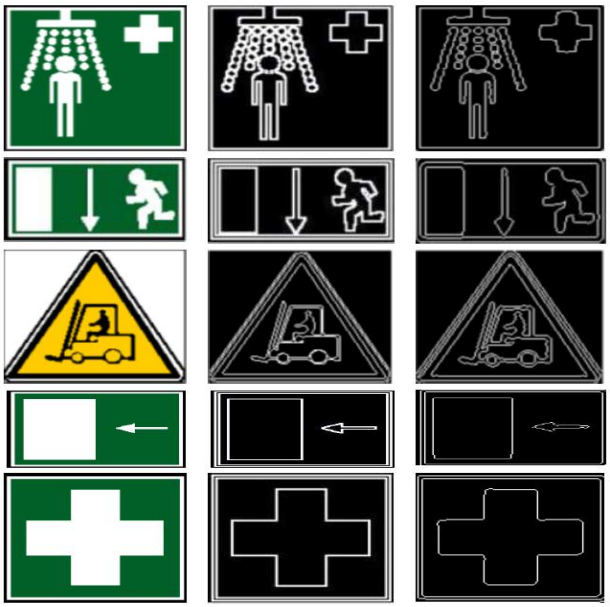

**Fig 3: from left to right: the initial image, the image segmented manually and the result of the proposed approach.**

The combination of operators having the highest quality of segmentation is (`wiener2,edge,bwareaopen`) and the most rewarded action is (`5;(prewitt,3.000000e-002);(10,8)`). It is the combination of operators decided by the agent OA and returned to the agent UA. Faced to any image from the same family (traffic signs), the user can execute directly this combination of operators with their parameters values. Making use of this result we segmented 30 images of traffic signs. Results are very satisfactory. Fig 4 shows the results of segmentation for three images from the same class of images (traffic signs) taken randomly from the processing.

Thus, our system finds among the proposed operators, the best ones with their optimal parameters values to apply and in which order. If the agent UA changes some information, like the set of the proposed operators for each phase, the final result changes also.

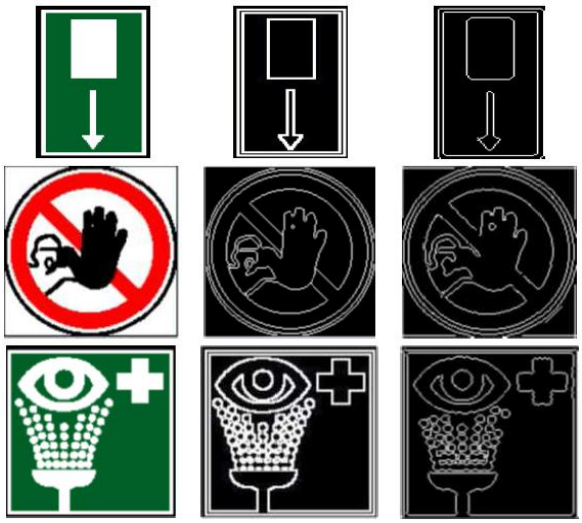

**Fig 4: from left to right: the initial image, the image segmented manually and the results of segmentation using the combination of operators founded by the agent OA**

Our approach constitutes a new general way of reasoning for any vision task that requires the right choice of operators and the right adjustment of their parameters.

## 5. CONCLUSION

Choosing the appropriate operators to apply and then adjusting their parameters values to accomplish a vision task represent a big challenge for users. In this paper we presented a multi-agents architecture based on reinforcement learning, which helps users by proposing them the optimal series of operators to apply and their best parameters values. Our system proceeds automatically to decide for his choices. Through the reinforcement learning mechanism, our architecture dose not considers only the system opportunities but also the user preferences. We intended to propose a general new way of thinking about the automatic selection of operators and the automatic adjustment of their parameters without the user intervention. The proposed approach constitutes then a theoretical robust basis for vision users and not just a solution for a particular problem. The experience we have done does not restrict the application of the approach to the image processing field, but its theoretical procedure shows that it can be applied to any decision process using parametric methods. Despite the theoretical strength of the idea and the obtained results, we acknowledge that we must improve the learning algorithm and study the reward expression using a function based on the similarity between the resulting images and the ground-truth.

## 6. REFERENCES

[1] R. Clouard, A. Elmoataz & F.Angot, "PANDORE : une bibliothèque et un environnement de programmation d'opérateurs de traitement d'images", Rapport interne du GREYC, Caen, France, Mars 1997.

[2] B.A. Draper, J. Bins, and K. Baek, "*ADORE*: Adaptive Object Recognition". Videre, 2000. 1(4): p. 86-99.

[3] B. Nickolay, B. Schneider, S.Jacob, "Parameter Optimization of an Image Processing System using Evolutionary Algorithms" 637-644. CAIP 1997.

[4] G. W.Taylor, "A Reinforcement Learning Framework for Parameter Control in Computer Vision Applications"

Proceedings of the First Canadian Conference on Computer and Robot Vision (CRV'04), IEEE 2004.

[5] F. Sahba, H. R. Tizhoosh, M. Salama. "Application of reinforcement learning for segmentation of transrectal ultrasound images" BMC Medical Imaging 2008, 8:8.

[6] I. Qaffou, M. Sadgal, A. Elfazziki:"A Reinforcement Learning Method to adjust Parameters of Vision Operators", Sixth International Conference on Intelligent Systems: Theory and Application. 23-29, Rabat 2010.

[7] I. Qaffou, M. Sadgal, A. Elfazziki:"A Reinforcement Learning approach to adjust parameters of texture segmentation", Wotic'09, Abstract p 53. Agadir 2009.

[8] RS. Sutton, AG. Barto: "Reinforcement Learning" Cambridge, MA: MIT Press; 1998.

[9] Watkins CJCH, Dayan P: "Q-Learning". Machine Learning 1992, 8:279-292.

[10] R Haroun. "Segmentation des tissus cérébraux sur des images par résonance magnétique". Master's thesis, Université des sciences et de la technologie Houari Boumediène, 2005.

[11] S.Chabrier, H.Laurent, C. Rosenberger, Y.J. Zhang, "Supervised evaluation of synthetic and real contour segmentation results", European Signal Processing Conference (EUSIPCO) 2006.

[12] DO Minh Chau, "Évaluation de la segmentation d'images". Rapport final TIPE. Institut de la francophonie pour l'informatique. Nanoï 2007.