Journal of Electronic Imaging

SPIEDigitalLibrary.org/jei

Evolutionary adaptive eye tracking for low-cost human computer interaction applications

Yan Shen
Hak Chul Shin
Won Jun Sung
Sarang Khim
Honglak Kim
Phill Kyu Rhee

SPIE

IS&T
imaging.org

Evolutionary adaptive eye tracking for low-cost human computer interaction applications

Yan Shen
Hak Chul Shin
Won Jun Sung
Sarang Khim
Honglak Kim
Phill Kyu Rhee
Inha University
235 Yong-Hyun Dong, Nam Ku
Incheon, Republic of Korea
E-mail: pkrhee@inha.ac.kr

Abstract. *We present an evolutionary adaptive eye-tracking framework aiming for low-cost human computer interaction. The main focus is to guarantee eye-tracking performance without using high-cost devices and strongly controlled situations. The performance optimization of eye tracking is formulated into the dynamic control problem of deciding on an eye tracking algorithm structure and associated thresholds/parameters, where the dynamic control space is denoted by genotype and phenotype spaces. The evolutionary algorithm is responsible for exploring the genotype control space, and the reinforcement learning algorithm organizes the evolved genotype into a reactive phenotype. The evolutionary algorithm encodes an eye-tracking scheme as a genetic code based on image variation analysis. Then, the reinforcement learning algorithm defines internal states in a phenotype control space limited by the perceived genetic code and carries out interactive adaptations. The proposed method can achieve optimal performance by compromising the difficulty in the real-time performance of the evolutionary algorithm and the drawback of the huge search space of the reinforcement learning algorithm. Extensive experiments were carried out using webcam image sequences and yielded very encouraging results. The framework can be readily applied to other low-cost vision-based human computer interactions in solving their intrinsic brittleness in unstable operational environments.*

1 Introduction

In the last three decades, many eye-tracking approaches have been proposed aimed at diverse objectives and applications.[1,2] Automatic eye tracking has been employed in many application areas, since it is strongly connected with human perception, attention, and cognitive states. Eye-tracking technology can be employed in unobtrusive and hands-free human computer interaction (HCI) as an effective tool of interaction and communication between computers and people. Even though a lot of effort has been invested in developing the technology, the problem of eye tracking is still very challenging, owing to changing operation environments with varying types of illumination, viewing angle, scale, individual eye shape, and jittering.[3] Most successful eye-tracking techniques in commercial areas pose either the requirement of a high-cost image capturing device (e.g., camera and lens) or very limited operation within a strongly controlled situation.[4,5] Recently, much interest in eye-tracking technology is being generated in the areas of HCI for web usability, advertising, smart TV, and mobile applications. However, the high cost or the limitation of conventional eye-tracking techniques can hardly be employed in low-cost commercial applications. Without lighting control and controlled situations, more available and general eye-tracking technology that reduces costs as well as simplifies the complicated initial user setup is necessary for a successful HCI application. Few researchers have tackled the more general vision technology for eye tracking in a loosely controlled environment with low-cost equipment.[6,7] Most techniques, however, are far from practical in current and coming vision-based HCI applications due to their intrinsic brittleness in changing image-capturing environments.

One promising direction is an adaptive selection of different eye-tracking algorithm structures and the control of associated thresholds/parameters considering environment changes. However, the decision of an optimal algorithm structure with its adaptable thresholds/parameters is a very complicated problem in the area of image processing and computer vision in general.[8] Some techniques for adaptive thresholds and flexible algorithm structures can be found in evolutionary algorithm (EA) approaches such as genetic algorithm (GA), genetic programming (GP),[9] and other evolutionary learning methods. The EA that evolves in a manner resembling natural selection can be employed to solve complex problems even if its creator does not fully understand the phenomenon.[10] However, the evaluation of a real-time system using the EA approach very often encounters a critical difficulty due to a significant amount of computation time coming from repetitive computations for many individuals over several generations. The high computational cost of the EA is a serious limiting factor. Another limitation of most evolutionary algorithms is their ambiguity in discriminating genotype and phenotype. In nature, the fertilized ovum cell

Paper 12404 received Nov. 13, 2012; revised manuscript received Jan. 29, 2013; accepted for publication Feb. 1, 2013; published online Mar. 1, 2013.

undergoes a complex process known as embryogenesis to become a mature phenotype.[11] The outward, physical manifestation of an organism is anything that is part of an observable structure, function, or behavior of a living organism. The observable characteristics of an organism internally coded, inheritable information is carried by the genetic code, the inherited instruction which carries acquired characters. An organism is usually exposed to sequences of events and reinforcements, and the same genotype encounters diverse environments where different behavioral actions are optimal. Learning is required in the whole lifetime of an individual that enforces a selection pressure, even though a correct phenotype from birth seems to be produced.[12] Some successful evolutionary approaches combined with learning technologies can be found in robot controlling areas.[13]

Most parameter control approaches require a precise model that describes interactions with environments. In many cases, however, it is very complicated or impossible to model the interaction for precise real-time task processing. In a real world situation, offline learning with an imprecise model frequently cannot produce an acceptable performance.[14] The reinforcement learning (RL) approach is one of the promising approaches to solve such problems, since it is based on "learning by experience." The RL approach is very effective in controlling a system by interacting with its external environment.[15] A bunch of model-free value-function-based RL algorithms have been studied, such as the Q-learning, SARSA, and AC methods.[12,16] The RL approach can optimize system parameters through interactions with a perceived environment.[17,18] An RL-based eye-tracking scheme can learn state-action probabilities, provide performance-adaptive functionality without requiring a system model, and assure the convergence to an optimal action. However, a very large amount of repetitive trial and error is necessary in order to find the optimal performance, because of the curse of a huge search space in the visual tracking problem. The enormous amount of learning time required for a low-cost eye tracking task in varying environments prohibits the RL algorithm to be employed, since a huge search space for deciding actions requires a high computation time for learning by experience.

We propose an evolutionary adaptive framework for a high-performance eye tracking scheme for a low-cost vision-based HCI. The framework for efficient and robust eye tracking behaves adaptively by combining the evolutionary and interactive learning paradigms. It devises a performance-adaptive eye tracking scheme where possible configurations of eye tracking are represented as genotype and phenotype control spaces in terms of algorithm structure and thresholds/parameters. The EA explores to find an optimal eye tracking genetic code in the genotype control space where an illumination environment measured by image quality analysis and represented by image context.[19,20] The EA can evolve relatively long-term behaviors than the reactive behaviors of the RL,[14] while the RL provides quick interactive adaptation in a real world environment. The EA performs mainly in the genotype learning using a pre-collected training set in an off-line fashion. The huge control space of algorithm structures and thresholds/parameters is partitioned in accordance with a perceived image context, and the EA determines a genetic code that fits for the perceived image context. The RL's main role is to organize the phenotype of the scheme interactively. The RL algorithm defines its internal environmental state from the partitioned genotype control space and mainly performs a quick and interactive adaptation. In this way, the proposed method perceives the changing environment of the real world and learns an effective algorithm structure and associated thresholds/parameters for optimal eye tracking performance. The major contributions of this paper are summarized as follows:

1. Instead of posing a high-cost image capturing device or very limited situation, a performance guarantee is achieved by formulating eye tracking into the dynamic control problem of optimizing an algorithm structure with associated thresholds/parameters.
2. Tackling the dilemma of the EA and the RL algorithms, i.e., the intrinsic non-real-time property of the EA and the curse of huge search space of the RL algorithm, the framework can efficiently improve the performance of the eye tracking scheme in terms of accuracy and speed. The EA approach can reduce the curse of high dimensionality and the huge number of trials of the RL algorithm, and accelerate the convergence speed of the RL algorithm compared with learning from scratch.
3. The proposed framework defines the RL internal environment in connection with the genotype control space instead of an input image as the RL environment in Refs. 8 and 21, which cannot fully utilize the interactive characteristics of the RL algorithm. While their approach can be thought to take an image sequence as stimuli, our approach associates the RL environment state with thresholds/parameters of the eye tracking scheme. As a result, the proposed framework can explore the consecutive and interactive threshold/parameter space influenced by a changing external environment, and thus fully take advantage of the RL algorithm by providing very flexible and high performance in the eye tracking scheme.
4. The framework can be readily applied to other vision-based HCI applications in solving their intrinsic brittleness under changing image capturing environments.

This paper is organized as follows. In Sec. 2, the evolutionary adaptive framework is discussed for controlling eye tracking parameter spaces. The genotype control space and evolution are discussed in Sec. 3. The phenotype manifestation using the RL algorithm is given in Sec. 4. In Sec. 5, the performance measures of eye tracking are discussed. Finally, the experimental results and concluding remarks are given in Secs. 6 and 7, respectively.

2 Evolutionary Adaptive Framework for Eye Tracking Parameter Control

The rationale behind the proposed eye tracking scheme with evolutionary adaptive capability is the recurrence phenomenon of external stimulus, here image quality variation influenced by the changes of the image-capturing environment mainly from illumination, viewing angle, and individual eye shape. The proposed scheme explores an optimal algorithm structure and associated threshold/parameter space guided by the observation of image variations that mainly affects scheme performance. The proposed scheme consists of three

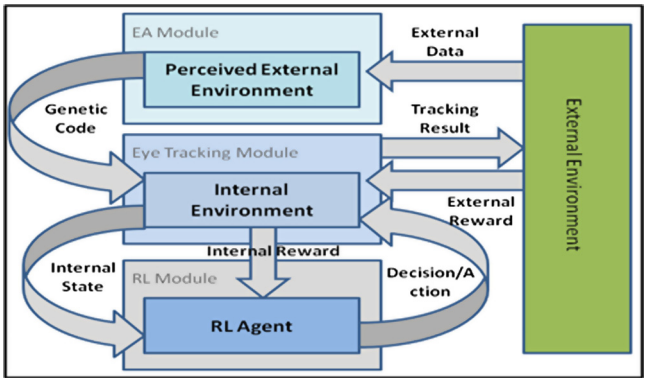


Fig. 1 The block diagram of the evolutionary adaptive eye-tracking framework.

modules, as shown in Fig. 1: the EA module, the eye-tracking module, and the RL module. It continuously perceives and interacts with the external environment, and adapts its algorithm structure, thresholds, and parameters. The proposed framework includes genotype evolution and phenotype adaptation. The first one performs long-term learning based on the EA, and the second one performs interactive phenotype learning using the RL algorithm.

The performance of an eye-tracking scheme is highly dependent upon the choice of algorithms in each step and their associated parameters and thresholds.[20] It is best formulated into a nontrivial dynamic control problem of deciding on an eye-tracking algorithm structure and associated thresholds and parameters. Considering the primary cause of degrading the eye-tracking performance is image quality variation due to illumination, pose, and eye shape, we introduce two level intelligent control mechanisms to optimize the eye-tracking scheme. The first level control mechanism employs EA for the genotype evolution, which determines a possible optimal scheme configuration for a perceived external environment as an image context, where an external environment measured by k-means clustering and image quality analysis.[19,20] The second one takes advantage of the RL algorithm for a phenotype manifestation of the genetic code.

The eye area is located using the face location and eye area location methods in Ref. 20, and eye tracking is processed based on the eye tracking method in Ref. 22. The initial eye centroid is estimated by preprocessing, feature extraction, and a partial Hough transformation. Preprocessing is carried out using an adaptively selected algorithm substructure and its associated thresholds, and feature extraction is performed to produce a contour or edge image. The contour or the edge image is processed by the partial Hough transform with adjusted arc angle parameters of the iris boundary to determine an optimal tracking point. Finally, the Kalman filter is applied, and the next eye centroid is predicted. The above steps are repeated until eye tracking is terminated. The eye-tracking control space, i.e., the possible algorithm structures and the ranges of thresholds/parameters, is determined based on prior knowledge or experiments considering the tradeoff between tracking accuracy and execution time constraints. The estimation steps of the eye centroid, the eye centroid prediction by the Kalman filter,[23,24] and their control space are discussed in the remainder of this section.

2.1 *Algorithm Structure and Thresholds/Parameters in Preprocessing and Feature Extraction*

The preprocessing step considered here consists of histogram equalization, Retnix with one threshold,[25] and end-in contrast stretching with two parameters.[26] For example, possible algorithm structures of the preprocessing step are histogram equalization, Retnix, Retnix with histogram equalization in a serial combination, or the end-in contrast stretching. The feature extraction step can be binarization, the Canny edge detection algorithm,[27] binarization and the Canny in parallel combination with an AND operation, or binarization and the Canny in parallel combination with an OR operation, where binarization and the contour in serial combinations is denoted by binarization for simplicity. Feature extraction is selectively carried out using one of the above four algorithm structures with different thresholds. Binarization has one threshold, and the Canny algorithm has two thresholds for edge detection, where the first is related to edge linking, and the second is related to finding the initial segments of the strong edges.

2.2 *Parameters of the Partial Hough Transform*

In our eye tracking system, the iris boundary is approximated by a circle since the proposed method aims at real-time performance using low resolution images. The algorithm detects the center of the eye (iris) by calculating the outer boundary of the iris image. We adopt a modified Hough transform method, called partial Hough transform, to extract features that are less affected by noise and eyelid occlusions. Two portions of the circle (partial circles) are matched instead of the entire circle to minimize the occlusion effects of the upper and lower eyelids. A four-dimensional control space for eye centroid tracking includes the angle values indicating the partial iris boundaries of interest, i.e., the partial circles (see Fig. 2). The partial Hough transform algorithm for circle

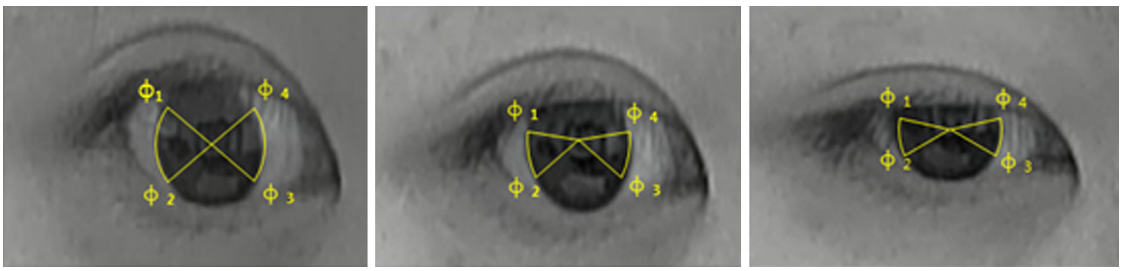


Fig. 2 Two partial circles of iris outer boundaries with different eye opening degrees: the portions of the outer iris boundary between the angles of ϕ_1 and ϕ_2, and that of ϕ_3 and ϕ_4 are considered instead of all iris boundaries avoiding eyelid occlusion effects.

fitting can then be described as follows. The iris outer circle can be represented as

$$(x-a_0)^2+(y-b_0)^2=r^2, \tag{1}$$

where (a_0, b_0) is the coordinates of the eye centroid (the center of outer iris boundary), and r is the circle radius.

The two stage algorithm formulated in Ref. 23 is employed: the first stage finds the eye circle center, and the second stage estimates normal directions to the outer iris boundary points on the partial circles intersecting the eye centroid. Let (x, y) be the gradient direction point on the outer iris boundary. A mapping from (x, y, ϕ) triples into the center parameters (a, b) produces a straight line, where ϕ is the angle of the gradient direction. The intersection of many of these lines identifies the coordinates of the eye centroid. The relation between (x, y, ϕ) and (a, b) is given as

$$\phi=\arctan\left[\frac{y-b}{x-a}\right]. \tag{2}$$

In the partial Hough transform algorithm, the range of the outer iris boundary is restricted by four angle parameters, ϕ_1, ϕ_2, ϕ_3, and ϕ_4, to avoid the effect of eyelid occlusion, i.e.,

$$\phi\in[\phi_1,\phi_2]\cup[\phi_3,\phi_4]. \tag{3}$$

2.3 *Noise Parameters of the Kalman Filter*

The Kalman filter[28] is a recursive approach for the discrete linear filtering problem by estimating a state process that minimizes the squared error.[29,30] In this paper, the Kalman filter is employed to approximate the tracking of eye centroids formulated as a Bayesian tracking problem.

Assuming that the motion of the eye centroid has a constant velocity F_t, the covariance matrices of the state transition model and the process noise model can be determined as follows:[29]

$$F_t=\begin{bmatrix}1&0&\Delta T&0\\0&1&0&\Delta T\\0&0&1&0\\0&0&0&1\end{bmatrix} \tag{4}$$

and

$$Q_{t-1}=\begin{bmatrix}0&0&0&0\\0&0&0&0\\0&0&q^2&0\\0&0&0&q^2\end{bmatrix}, \tag{5}$$

where ΔT is the time interval between two adjacent frames that is usually very short, and q is a parameter related to the process nose. Assuming that H_t is constant, the matrices of the measurement model and the measurement noise covariance are determined, respectively, as follows:

$$H_t=\begin{bmatrix}1&0&0&0\\0&1&0&0\end{bmatrix} \tag{6}$$

and

$$R_t=\begin{bmatrix}r^2&0\\0&r^2\end{bmatrix} \tag{7}$$

where r is a parameter related to the measurement noise. The Kalman filter approach often encounters difficulty in estimating the noise covariance matrices Q_{t-1} and R_t. In many applications, the noise covariance's Q_{t-1} and R_t will be stabilized quickly and remain constant.[31] The parameters r and q can be pre-computed by running the filter offline or determining the state-state values.[32] However, the noise does not remain constant in eye tracking due to the uncertainties of dynamically changing environments. In this context, the parameters r and q are evolved by the EA as discussed in this session and adapted by the RL method in the next section in order to achieve optimal performance in our eye-tracking scheme.

3 Genotype Evolution

In this section, we discuss how to encode the variation of the algorithm structure and associated thresholds and parameters of the eye-tracking scheme into a genetic code so that the genotype control space is explored for evolutionary optimization. The genotype of the tracking scheme is denoted by genetic codes in the genotype control space, each of which represents an algorithm structure and thresholds/parameters, and thus an optimal algorithm configuration that has evolved for a given external environment is denoted by a genetic code. The algorithm structure of the tracking scheme is divided into the preprocessing, feature extraction, partial Hough transform, and Kalman prediction as discussed previously. Given an external environment, an optimal algorithm structure with associated thresholds/parameters is determined in accordance with a perceived image context.

In general, a context can be any information that affects the performance of a system of interest. Image context might be affected by lighting direction, brightness, contrast, and spectral composition. The concept of an image context with image quality analysis technology can improve system performance.[33] Recently, image quality analysis methods have been successfully applied to diverse applications such as image storage, compression, communication, display, segmentation, and recognition,[34] as well as used to decide selective preprocessing for an adaptive illumination normalization using adaptive weighting parameters.[19,20] In this paper, an image quality analysis is adopted for the categorization of the variation of input image quality that is influenced by the external environment. For simplicity of discussion we employ only input image illumination changes as a clue to perceive the changes of an external environment. The Haar wavelet-based Bayesian classification is performed for face and eye area detection[20] before eye tracking begins.

Each substructure is constructed by combining action units. An algorithm structure can be divided into the substructure sequence, and each substructure is configured by the combination of action units which are the basic building blocks of the system. Let Z be the set of thresholds and/or parameters of an action unit (AU). Each action unit has associated thresholds and/or parameters if they are required, and are denoted as follows:

$$\mathrm{AU}_k(Z_k)=\mathrm{AU}_k(\theta_{k_1},\ldots,\theta_{k_n}), \tag{8}$$

where θ_{k_i} is the i'th threshold/parameter of action unit AU_k. Let Ψ_A be a configurable algorithm substructure, say A. It is denoted as

Table 1 An illustration of the genotype encoding format of the proposed eye tracking scheme and the parameter ranges.

	Algorithm substructure													
	Preprocessing				Feature extraction				Partial Hough transform				Kalman	
Ψ/θ	Ψ_{Pre}	θ_{RX}	θ_{ECS1}	θ_{ECS2}	Ψ_{FE}	θ_{BN}	θ_{CN1}	θ_{CN2}	θ_{PHT1}	θ_{PHT2}	θ_{PHT3}	θ_{PHT4}	θ_{KF1}	θ_{KF2}
No. of bits	2	4	4	4	2	4	4	4	4	4	4	4	4	4
R_{min}		181	32	160		68	4	32	148	230	278	0	0.005	0.05
R_{max}		245	96	224		188	20	64	180	262	310	32	0.015	0.15

Note: R_{min} and R_{max} indicate the minimum and maximum of the range values of each parameter, respectively.

$$\Psi_A = \Psi_A[\mathrm{AU}_1^A(Z_1^A), \ldots, \mathrm{AU}_a^A(Z_a^A)]. \tag{9}$$

It means that Ψ_A is configured from action units $\mathrm{AU}_1^A, \ldots, \mathrm{AU}_a^A$. In the eye-tracking scheme, the whole algorithm structure is divided into four consecutive substructures, i.e., the preprocessing, feature extraction, partial Hough transform, and Kalman filter prediction. The preprocessing has three action units histogram equalization (HE), Retnix (RX), and end-in contrast stretching (ECS). Feature extraction is represented by action units binarization (BN) and Canny algorithm (CN). Finally, eye tracking is performed by action units partial Hough transform (PHT) and Kalman filter (KF). Then, the whole algorithm structure of our eye-tracking scheme is described as follows:

$$\begin{aligned}\Psi &= \Psi_{\mathrm{Pre}} \gg \Psi_{\mathrm{FE}} \gg \Psi_{\mathrm{PHT}} \gg \Psi_{\mathrm{KF}} \\ &= \Psi_{\mathrm{Pre}}[\mathrm{HE}, \mathrm{RX}(\theta_{\mathrm{RX}}), \mathrm{ECS}(\theta_{\mathrm{ECS1,ECS2}})] \\ &\gg \Psi_{\mathrm{FE}}[\mathrm{BN}(\theta_{\mathrm{BN}}), \mathrm{CN}(\theta_{\mathrm{CN1}}, \theta_{\mathrm{CN2}})] \\ &\gg \Psi_{\mathrm{PHT}}[\mathrm{PHT}(\theta_{\mathrm{PHT1}}, \theta_{\mathrm{PHT2}}, \theta_{\mathrm{PHT3}}, \theta_{\mathrm{PHT4}})] \\ &\gg \Psi_{\mathrm{KF}}[\mathrm{KF}(\theta_{\mathrm{KF1}}, \theta_{\mathrm{KF2}})],\end{aligned} \tag{10}$$

where Ψ_{Pre}, Ψ_{FE}, Ψ_{PHT}, and Ψ_{KF} denote the algorithm substructures of the preprocessing, feature extraction, partial Hough transform, and Kalman filter prediction, respectively, and θ_{RX} denotes the scale parameter of action unit RX, and so on. Let image context be denoted as L_t at time-step t, then the eye tracking algorithm configuration is illustrated as follows:

$$\begin{aligned}\Psi(L_t) &= \Psi_{\mathrm{Pre}}(L_t) \gg \Psi_{\mathrm{Fe}}(L_t) \gg \Psi_{\mathrm{PHT}}(L_t) \gg \Psi_{\mathrm{KF}}(L_t) \\ &= \Psi_{\mathrm{Pre}}[\mathrm{HE} \gg \mathrm{RX}(\theta_{\mathrm{RX}} = \theta_1)] \\ &\gg \Psi_{\mathrm{FE}}[[(\mathrm{CN}(\theta_{\mathrm{CN1}} = \theta_3, \theta_{\mathrm{CN2}} = \theta_4)]\|_{\mathrm{AND}}\mathrm{BN}(\theta_{\mathrm{BN}} = \theta_2)] \\ &\gg \Psi_{\mathrm{PHT}}[\mathrm{PHT}(\theta_{\mathrm{PHT1}} = \theta_5, \theta_{\mathrm{PHT2}} = \theta_6, \theta_{\mathrm{PHT3}} = \theta_7, \\ &\quad \theta_{\mathrm{PHT4}} = \theta_8)] \\ &\gg \Psi_{\mathrm{KF}}[\mathrm{KF}(\theta_{\mathrm{KF1}} = \theta_9, \theta_{\mathrm{KF2}} = \theta_{10})],\end{aligned} \tag{11}$$

where the preprocessing of the configured eye tracking algorithm consists of the HE followed by the RX with threshold θ_1; feature extraction consists of a parallel combination using the AND operation between CN with thresholds θ_3, θ_4, and BN with threshold θ_2, and so on. Genotype encoding format of the eye tracking system in the genotype control space is illustrated Table 1. We assign the preprocessing algorithm substructure 2 bits since the feasible combinations are four types, i.e., HE, RX, ECS, and HE followed by RX (HE $\gg$ RX). The feature extraction algorithm substructure is assigned 2 bits where the feasible combinations are BN, CN, the parallel combination of BN and CN using the AND operation, and the parallel combination of BN and CN using the OR operation. Since the partial Hough transform and the Kalman prediction have only one algorithm structure, no bit is assigned as the algorithm structure parameter.

Table 2 Four types of algorithm substructure denoted by its 2-bit vector of the genetic code in the feature extraction step.

Bit vector of Ψ_{FE}	00	01	10	11
Feature extraction algorithm substructure	Binarization	Canny	Binarization $\|\|_{AND}$ Canny	Binarization $\|\|_{OR}$ Canny

If the feature extraction step is determined to have four types of algorithm structure, it is denoted by its 2-bit vector of the genetic code as shown in Table 2.

Let θ_A be a threshold (parameter) of an action unit A. The pivot $\theta_A^{\mathrm{pivot}}$ is defined as the central value of a threshold range of θ_A in the phenotype control space. Let $\theta_{A\,\mathrm{min}}$ and $\theta_{A\,\mathrm{max}}$ denote the minimum and the maximum threshold values of θ_A, respectively. The interval of the threshold pivots between adjacent bit vector, $\theta_{A\alpha}$ is calculated by

$$\theta_{A\alpha} = \frac{\theta_{A\,\mathrm{max}} - \theta_{A\,\mathrm{min}}}{2^\beta - 1}, \tag{12}$$

where β is the number of bits representing θ_A (see Table 1). The pivots of the action unit A are derived by adding $\theta_{A\alpha}$ starting from $\theta_{A\,\mathrm{min}}$ up to $\theta_{A\,\mathrm{max}}$. For example, if $\theta_{\mathrm{BN\,min}}$ and $\theta_{\mathrm{BN\,max}}$ are assumed to be 68 and 188 in gray level, respectively, the lookup table for the binarization threshold pivots of a sub-genetic code of θ_{BN} are illustrated in Table 3.

Genotype evolution is performed to find an optimal genetic code for each image context. The fitness is calculated by the average tracking rate [Eq. (30)] discussed in Sec. 5.

4 Phenotype Manifestation Using the RL Algorithm

As discussed in the previous section, the EA evolves the system configuration in the dynamic control space of the

Table 3 An illustrative lookup table of the binarization threshold pivots of individual sub-genetic code of θ_{BN}.

K (subgenetic code of θ_{BN})	$\theta_{BN}^{pivot}(K)$ binarization pivot	K (subgenetic code of θ_{BN})	$\theta_{BN}^{pivot}(K)$ binarization pivot
0000	68	1000	132
0001	76	1001	140
0010	84	1010	148
0011	92	1011	156
0100	100	1100	164
0101	108	1101	172
0110	116	1110	180
0111	124	1111	188

algorithm structure and its associated thresholds/parameters in terms of genotype representation so that they are put together into the eye tracking scheme to optimize performance. The role of the EA is to find an appropriate scheme genotype, i.e., a genetic code representing an optimal eye tracking configuration based on a perceived external environment, (i.e., a perceived image context here) and the fitness evaluation. Since real world external environment changes are not fully predictable in the EA evolution step, the RL algorithm is employed to seek a precise phenotype manifestation of eye tracking in terms of thresholds/parameters and the reward. Combining the EA evolution and the RL adaptation capabilities, not only can the curse of huge control space of possible eye tracking configurations in applying the RL algorithm be handled, but also the difficulty of the EA in a real-time adaptation.

In our approach, the dynamic control space of the EA is not the same as that of the RL algorithm, but they are interrelated through the performance optimization in a real world environment. A genetic code, generated by the EA for each image context, defines a related phenotype control space which describes by the ranges of thresholds and/or parameters. For simplicity and real-time consideration, the algorithm structure in a genetic code will not be changed in the phenotype manifestation, and it is also intuitively suitable with a real world evolutionary adaptation phenomenon. The RL algorithm treats the adaptation of the thresholds/parameters as a consecutive decision process, i.e., the RL state transition, to obtain an optimal configuration of eye tracking. Regarding the eye-tracking optimization as the events of consecutive trials for deciding the RL state observed in discrete time, a Markov decision process (MDP) model which is necessary to use the RL algorithm need to be justified. In theory, the RL algorithm cannot be applied exactly if the Markov property is not satisfied. However, the RL application developed based on the Markov assumption is still valid in many cases, to approximate and understand system behavior. Even when a state signal does not satisfy the Markov property, the RL algorithm can be employed in more complex and realistic non-Markov cases, and the RL algorithm has been successfully applied in many cases by approximating the state as a Markov state.[16]

In general, an MDP approach for a RL algorithm requires five-tuples: states, actions, transition function, rewards and discount factor.[35] The action can be any decision/behavior that the RL agent needs learn, and a state can be any factor that influences the agent's decision making. In this paper, each phenotype manifestation, denoted by the phenotype control space of associated thresholds/parameters defines a state in the RL internal environment (see Fig. 1), and interacts with the RL agent to explore an optimal threshold/parameter set maximizing the eye-tracking performance. Here, the action is defined as the move to a next state that decides next phenotype manifestation in the phenotype control space.

The produced action activates a state transition of the internal environment from the current internal state in the phenotype control space into a new state, and the RL agent receives a reinforcement signal from the internal environment. One can notice that the proposed approach is distinguished from Refs. 8 and 21 where an image itself is modeled as an RL state, and the parameter decision is as an action. Their approach can be thought to take an image sequence as stimuli and to produce a segmented image. They cannot fully utilize the interactive characteristics of the RL algorithm in full, since the agent can alter the state at each time-step by taking actions.[36] On the other hand, our approach relies on consecutive and interactive threshold/parameter transitions influenced by changing external environments (see Fig. 1), which are modeled as a Markov decision process. In this paper, we will stay with the concepts of environment, agent, reward, and action in a usual RL. However, we use the terms internal environment and internal reward to avoid a possible confusion between external environment and external feedback. On the other hand, the internal environment can be interpreted as an intrinsic motivation used in cognitive science, where intrinsically motivated behavior is emphasized as a vital factor for intellectual growth.[37] RL action can also be interpreted as internal decision instead of an action as a human's decision to move his muscles in a certain direction.

It can be interpreted that the next values of thresholds/parameters are mainly decided by the current internal state, i.e., the current values of thresholds/parameters of the tracking scheme. The RL internal states are members of a finite set denoting all possible eye tracing scheme configurations in terms of associated thresholds/parameters in the phenotype control space, which is related and limited by a genetic code influenced by the external environment (see Fig. 1). Thus, the RL internal state transits randomly from one state to another within the limited control space in discrete time-steps.

In the RL algorithm, an episode is defined to separate the agent-environment interaction into subsequences. An episode can be thought as a temporal unit of repeated interactions, and an episode has a standard starting state and a terminal state.[16] The RL starting state is declared either when the eye-tracking confidence is degraded below a predefined criterion, say the RL starting threshold η, or an alert signal is received from the external environment through the external feedback (see Fig. 1). The RL terminal state is

declared when the eye-tracking confidence reaches a predefined criterion, say RL terminal threshold ζ, or it does not improve any more. The current internal state is the estimation of the time-discounted amount of the reward to be expected starting from that internal state and continuing to act according to its policy.

Considering real world constraints on computation resources, the set of internal states in the phenotype control space limited by a genetic code determined by a perceived image context

$$S = \{s_1, \ldots, s_k, \ldots, s_K\}, \tag{13}$$

where s_k is a state which indicates the indexes of the threshold/parameter lookup table in the eye tracking scheme. In other words, all possible phenotype manifestations of the eye tracking scheme are indexed by internal environment states. Let the dimension of the phenotype control space be p. Then, an internal environmental state at time-step t is denoted as

$$s_t = [\theta_{t,1}, \theta_{t,2}, \ldots, \theta_{t,p}], \tag{14}$$

where $\theta_{t,i}$ is an index of the lookup table representing an associated threshold/parameter at time-step t (e.g., Table 4).

A finite set of actions is available in each internal state, and $\alpha_t \in A(s_t)$ indicates the decision action of changing the thresholds/parameters of the eye tracking scheme at time-step t. Let $a_t = [\zeta_{t,1}, \zeta_{t,2}, \ldots, \zeta_{t,p}]$ be an action at time-step t, where $\zeta_{t,i}$ is $-d_{t,i}$, 0, or $+d_{t,i}$ meaning that the negative direction movement of d units, stationary, or positive direction movement of d units in the i'th coordinate (index) of the phenotype control space at time-step t.

Given threshold/parameter values, the tracking scheme will generate possible eye centroids within a search window. Let $s_t = [\theta_{t,1}, \theta_{t,2}, \ldots, \theta_{t,p}]$ be a phenotype control vector at time-step t, where the dimension of the threshold/parameter space is p and $\theta_{t,i}$ is an RL index of the lookup table representing a threshold or a parameter. The index has the discrete range precision. For example, θ_{BN} has the discrete range precision "8" in Table 4, meaning that the index value of θ_{BN} is one of the values in $\{0, 1, \ldots, 7\}$. Table 4 illustrates the state vector of the proposed eye tracking scheme with its discrete range precision.

Recall that the pivot $\theta_A^{\mathrm{pivot}}$ is defined as the central value of a threshold range of θ_A in the phenotype control space. The possible thresholds (parameters) of a RL state are decided in the RL lookup table as illustrated in Table 5 for the binarization threshold. Let χ be the interval between adjacent RL thresholds (parameters) for $\theta_A^{\mathrm{pivot}}$. The RL lookup table is generated using the following equation:

$$\mathrm{RL}(i, \theta_A^{\mathrm{pivot}}) = \left\{\theta_A^{\mathrm{pivot}}(K) - \left[\frac{\chi}{2}\right]\right\} + \left\{i - \left[\frac{v(\theta_A)}{2} - 1\right]\right\} \times \chi, \tag{15}$$

where i is the RL index of threshold (parameter) $\theta_A^{\mathrm{pivot}}$, K is a sub-genetic code, and $v(\theta_A)$ is the discrete range precision of θ_A (see Table 4). As an example, if the binarization sub-genetic code is decided as "0110," then the binarization threshold pivot of the genetic code is "116" in gray level intensity as shown in Table 3. Table 5 shows the RL lookup table of the binarization discrete range in the phenotype control space and dedicated RL thresholds which are denoted by intensity values. For example, the RL threshold when $i = 0$ and $\chi = 2$ is calculated as follows: $\mathrm{RL}(0, \theta_{\mathrm{BN}}) = \{116 - 1\} + \{0 - (8/2 - 1)\} \times 2 = 109$. If the RL state has a binarization RL index of "4," binarization is performed using the gray value 117 as the threshold in Table 5.

The internal reward function which is indirectly influenced by the external feedback defines the goal in our reinforcement learning (see Fig. 1). It maps each internal environment state and action pair to an expected internal reward. The reinforcement learning specifies how the agent alters its threshold and/or parameter decision policy as a result of its experience in eye tracking. The goal is to maximize the total amount of expected internal rewards over the long run. In this context, the immediate reward of the proposed RL algorithm is defined as follows: if the eye tracking is successful using Eq. (29) for each image frame, the high score is returned as the internal reward. Otherwise, the low score is returned as the reward where the high score and the low score are determined experimentally.

The aim of the RL algorithm is to optimize its long-term performance using the feedback of one-step immediate performance of eye tracking. We choose a temporal difference

Table 4 An example of the RL internal state format each element of which is a threshold/parameter within its discrete range precision.

Substructure	Preprocessing			Feature extraction			Partial Hough transform				Kalman	
Threshold/parameter	θ_{RX}	θ_{ECS1}	θ_{ECS2}	θ_{BN}	θ_{CN1}	θ_{CN2}	θ_{PHT1}	θ_{PHT2}	θ_{PHT3}	θ_{PHT4}	θ_{KF1}	θ_{KF2}
Discrete range precision	8	8	8	8	8	8	4	4	4	4	16	16

Table 5 An illustration lookup table for the phenotype discrete range of the binarization bit vector of genetic code "0110" and its dedicated thresholds in intensity values when the genotype binarization threshold is determined as "116."

i (RL index)	0	1	2	3	4	5	6	7
$\mathrm{RL}(i, \theta_{\mathrm{BN}})$	109	111	113	115	117	119	121	123

(TD) learning approach since it can provide an online, fully incremental learning capability, and since the TD approach learns from each transition regardless of what subsequent actions are taken.[16] The TD method also learns directly by experience without any knowledge of environment dynamics such as the model of its reward and next-state probability distributions. Two popular TD algorithms are SARSA and Q-learning, and they are different only in the estimation policy, i.e., on-policy and off-policy. Considering the real-time requirement of eye tracking, Watkins's one-step Q-learning[12] is selected which is relatively faster to converge than SARSA.[16] The one-step Q-learning algorithm estimates Q^* which makes the action-value functions, called Q-functions, and is an important algorithm of reinforcement learning with its simplicity, effectiveness, and model-free characteristic.[38] The one-step Q-learning method is a common off-policy temporal difference control algorithm. Q-learning accumulates optimal actions by evaluating action–value $Q(s,a)$ to construct sate-action table. The action-value updating equation of the one-step Q-learning is given as follows:[12]

$$Q_{t+1}(s,a) = Q_t(s,a) + \alpha[r_{t+1} + \gamma \max_{a'} Q_t(s',a') - Q_t(s,a)], \tag{16}$$

where $\alpha \in (0,1]$ is a learning rate and $\gamma \in (0,1]$ is a discounting rate. Action-value function Q learns and approximates an optimal action-value function, say Q^*, independent of the policy which enables early convergence with dramatic analysis simplification.[16,36]

The policy is the rules for selecting actions given the values of the possible next states, and the greedy policy is deterministic and takes the action with the maximum Q-value for every state, i.e.,

$$\pi(s) = \arg\max_a Q(s,a), \tag{17}$$

where $\pi\colon S \to A$ is deterministic. Equation (16) is valid only when we can find a stable reward for individual action-state pairs in our eye tracking scheme. However, the reward can only be consistent when the image quality of consecutive frames is near homogeneous. The assumption of near homogenous is not valid if the prior knowledge of image context from the EA module is disturbed due to a new external environment occurrence. To absorb the temporal uncertainty of an external environment, we employ a cooperative multi-agent RL[36] as follows: the joint action set $A = A_1 \times \cdots \times A_i \times \cdots \times A_n$, where A_i is the set of actions of the j'th agent, the state transition probability function $f\colon S \times A \to S$, and the reward function $r_i\colon S \times A \times S \to \Re$. The Q-learning equation and the greedy policy at time-step t are defined as follows:

$$Q_{t+1}(s,a) = Q_t(s,a) + \alpha[r_{t+1} + \gamma \max_{a'} Q_t(s',a') - Q_t(s,a)] \tag{18}$$

$$\pi_t(s) = \arg\max_{a_i} \max_{a_1,\cdots,a_{i-1},a_{i+1},\cdots,a_n} Q(s,a), \tag{19}$$

where $s \in S$ is a current state, action group $a = \{a_1, \cdots, a_n\}$ with $a_i \in A_i$ current action of the j'th agent, $s' \in S$ a next state, and a' a next action group. Each agent is an independent decision maker. The greedy does not explore the consequences of untried action sequences since it does not allow learning a current low-value action, but one that might lead to high value in the future. In this context, the ε-greedy policy is used to balance the actions between exploration and exploitation.[16] Our multi-agent Q learning algorithm is given in Algorithm 1.

Algorithm 1 Multi-agent Q-learning for eye tracking.

Repeat

Step 1. Choose a from s using ε-greedy policy.

Step 2. Take the actions, observe an immediate reward r by calculating the internal reward (see Algorithm 2), and observe the new internal state s'.

Step 3. Calculate the following:
$Q_{t+1}(s,a) = Q_t(s,a) + \alpha[r_{t+1} + \gamma \max_{a'} Q_t(s',a') - Q_t(s,a)]$

Step 4. $s \leftarrow s'$

Until the step limitation per frame M is reached or the success tracking criteria is satisfied [see Eq. (29)].

The whole evolutionary adaptive process for low-cost eye tracking is given in Algorithm 2. The framework learns, through repeated interactions with each other, to autonomously optimize eye-tracking performance at run-time. The limited system memory and the real-time delay constraints of the eye-tracking complementary accelerate the learning algorithm that exploits the perceived environmental knowledge about a system's external dynamics.

Algorithm 2 Evolutionary adaptive eye tracking.

Repeat

Step 1. Perceive the image context.

Step 2. Find an optimal genetic code for the perceived image context, decide the phenotype control space, and initialize all $Q(s,a)$ values arbitrarily.

Step 3. Repeat for consecutive image frames

Step 3.1. Perform Q-learning by calling Algorithm 1and decide the optimal thresholds/parameters.

Step 3.2. Perform the eye tracking until $\tau > \zeta$.

Step 4. If $\tau > \eta$, go to Step 3.

Step 5. If $\tau \le \delta$ (i.e., the eye tracking confidence τ is below the EA restarting threshold frequently where $\delta(< \eta)$, go to Step 1.

Until termination.

5 Performance Measures of Eye Tracking from Image Sequences

This section discusses the offline measures of eye-tracking performance such as recall, precision, harmonic mean, and the real-time measure, called real-time tracking confidence τ. Given N image frame sequence in a video be denoted as $V = (I_1, I_2, \ldots, I_N)$, the sequence of ground truth iris area will be compared with that of tracked iris areas for offline measures. Let $X = (x_1, x_2, \ldots, x_N)$ and $Y = (y_1, y_2, \ldots, y_N)$ be the sequence of ground truth iris areas and that of tracked iris areas, respectively. Owing to the evaluation method from information retrieval research, tracking performance can be measured by recall and precision.[39] The overlapped area divided by the union of the ground truth and tracked (detected) iris area is used for measuring a successful tracking in an image frame I_i.[40] The inner circle area of the iris is excluded to prohibit the effect of possible light source reflection as shown in Fig. 3.

$$\vartheta_i = \frac{\text{Area}(x_i \cap y_i)}{\text{Area}(x_i \cup y_i)}. \tag{20}$$

The constraint for successful tracking is defined as

$$\vartheta_i > t_m, \tag{21}$$

where t_m is a matching threshold and determined experimentally. In a general object tracking,[40] a tracking is considered to be correct if its overlap with ground truth bounding box is larger than 50%, i.e., $t_m = 0.5$. In this paper, t_m is decided as 0.6 since we require a more strict constraint for measuring the successful eye tracking.

Regarding the situations where a ground truth iris area cannot be defined due to heavy noise, and a tracked iris area cannot be found due to a shortage of the tracking algorithm, we define $\|\cdot\|$ operations as follows:

$$\|x_k\| = \begin{cases} 1 & \text{if } x_k \text{ exists in frame } I_k \\ 0 & \text{if } x_k \text{ does not exist in frame } I_k \end{cases} \tag{22}$$

$$\|y_k\| = \begin{cases} 1 & \text{if } y_k \text{ exists(can be detected) in frame } I_k \\ 0 & \text{if } y_k \text{ does not exist in frame } I_k \end{cases} \tag{23}$$

Then, the recall and precision measures of the tracked eye image sequence are defined as

$$\text{RE}(X, Y, t_m) = \frac{\sum_{k=1}^{N} f(x_k, y_k, t_m)}{\sum_{k=1}^{N} \|x_k\|} \tag{24}$$

$$\text{PR}(X, Y, t_m) = \frac{\sum_{k=1}^{N} f(x_k, y_k, t_m)}{\sum_{k=1}^{N} \|y_k\|}, \tag{25}$$

where f is the matching function, taking into account a successful track, i.e.,

$$f(x_k, y_k, t_m) = \begin{cases} 1 & \text{if } x_k \cap y_k \text{ matches to } x_k \cup y_k, \text{i.e., } \vartheta_k > t_m \\ 0 & \text{if the overlapped area } (x_k \cap y_k) \text{ does not match} \end{cases} \tag{26}$$

The harmonic mean (HM) of recall and precision which emphasizes the minimum of the two performance values is defined as follows:

$$\text{HM} = 2\frac{\text{RE}(X, Y, t_m) \times \text{PR}(X, Y, t_m)}{\text{RE}(X, Y, t_m) + \text{PR}(X, Y, t_m)}. \tag{27}$$

Since it is unrealistic to mark all ground truth iris areas in a real-time application, the real-time tracking confidence is defined and measured as follows. Let x_i' be the boundary area of a partial iris decided by the Hough transform and partial circle ranges as discussed in Sec. 2 and y_i' be the actual number of boundary iris pixels detected in the area. Given $V = (I_1, I_2, \ldots, I_N)$, let $X' = (x_1', x_2', \ldots, x_N')$ and $Y' = (y_1', y_2', \ldots, y_N')$. The real-time tracking mean (RM) at the time-step $k = q$ (i.e., at image frame I_q) is defined as the mean of successfully tracked adjacent frames between the time-steps $k = q - \delta$ delta and $k = q + \delta$, where δ is a parameter used for control the smoothness of the RM.

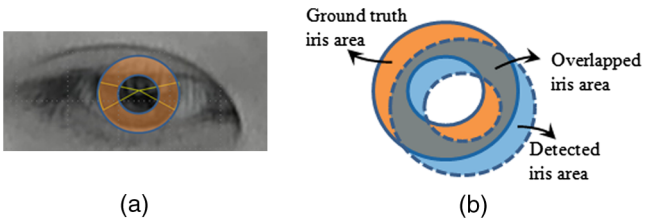


Fig. 3 Ground truth and overlapped iris areas for measuring a successful tracking, where the inner circle area of the iris is excluded to prohibit the effect of possible light source reflection in the center of iris: (a) the ground truth model of the iris, and (b) the overlapped iris area definition as the union of the ground truth and the detected iris areas.

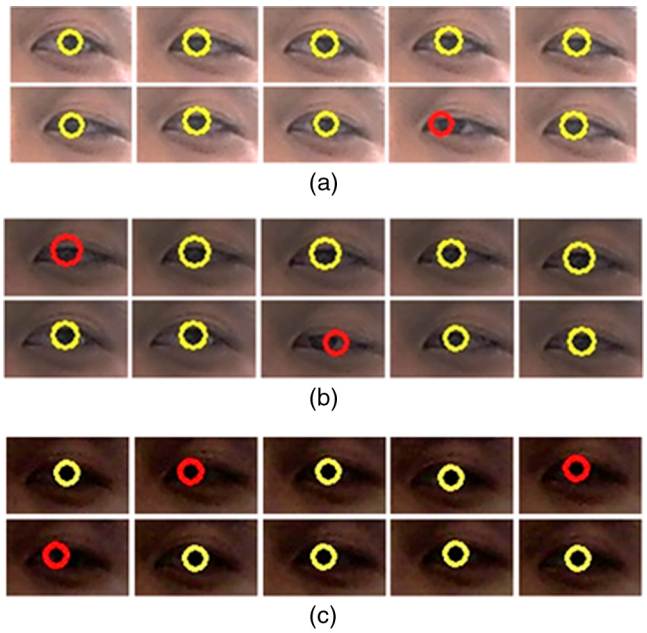


Fig. 4 Some examples of image frames used in analyzing the effects of image quality variance of the tracking system: (a) images with good quality illumination, (b) images with moderate quality illumination, and (c) images with bad quality illumination, where yellow circles indicate the successes in eye tracking and red circles indicate the failures.

The effect of different δ is discussed in Sec. 6.2. The RM is then calculated using the frame sequence $I_{q-\delta}, I_{q-\delta+1}, \ldots, I_{q+\delta-1}, I_{q+\delta}$ as follows:

$$\mathrm{RM}(X', Y', I_q, t'_m) = \frac{\sum_{k=q-\delta}^{k=q+\delta} g(x'_k, y'_k, t'_m)}{\sum_{k=q-\delta}^{k=q+\delta} \|x'_k\|}, \tag{28}$$

where the success of eye tracking at image frame I_q is defined using the constraint of the real-time matching threshold t'_m as follows:

$$g(x', y', t'_m) = \begin{cases} 1 & \text{if } \frac{y'_k}{x'_k} \geq t'_m \\ 0 & \text{otherwise} \end{cases}, \tag{29}$$

where t'_m is determined experimentally as 0.625 based on the threshold determination method[41] and the details can be found in Sec. 6.2.

Similarly, the average tracking rate (AR) for the whole video sequence $V = (I_1, I_2, \ldots, I_N)$ is defined as follows:

$$\mathrm{AR}(X', Y', I_q, t'_m) = \frac{\sum_{k=1}^{k=N} g(x'_k, y'_k, t'_m)}{\sum_{k=q-\delta}^{k=q+\delta} \|x'_k\|}. \tag{30}$$

However, Eq. (28) still cannot be used for real-time tracking performance estimation since at the time-step of image frame $k = q$, we cannot predict the tracking successes in image frames between $k = q+1, \ldots, k = \delta$ in prior. Thus, the real-time tracking confidence τ at the time-step of image frame I_q is estimated as the approximation of RM as follows:

$$\tau = \bar{\mathrm{RM}}(X', Y', I_q, t'_m) = \frac{\sum_{k=q-2\delta}^{k=q} g(x'_k, y'_k, t'_m)}{\sum_{k=q-2\delta}^{k=q} \|x'_k\|}. \tag{31}$$

6 Computational Experiments

We have tested the tracking performance of the single agent Q-learning and multiagent Q-learning algorithms using the evolutionary adaptive eye tracking framework. The EA module decides the phenotype algorithm structure and genetic code using the training set of image sequences, where the phenotype algorithm structure of the preprocessing is determined as the histogram equalization, and that of the preprocessing is determined as binarization. Thus, the tracking algorithm structure is determined as the histogram equalization in the preprocessing, binarization in the feature extraction, the partial Hough transform, and the Kalman filter in the EA module. That is, the phenotype algorithm structure is determined and represented as

$$\begin{aligned} \Psi = \Psi_{\mathrm{pre}}(\mathrm{HE}) &\gg \Psi_{\mathrm{FE}}[\mathrm{BN}(\theta_{\mathrm{BN}})] \\ &\gg \Psi_{\mathrm{PHT}}[\mathrm{PHT}(\theta_{\mathrm{PHT1}}, \theta_{\mathrm{PHT2}}, \theta_{\mathrm{PHT3}}, \theta_{\mathrm{PHT4}})] \\ &\gg \Psi_{\mathrm{KF}}[\mathrm{KF}(\theta_{\mathrm{KF1}}, \theta_{\mathrm{KF2}})]. \end{aligned}$$

The noise parameters in the Kalman filter are fixed after the EA processing for simplicity, and the experiments mainly focus on the interactive adaptation of θ_{BN}, θ_{PHT1}, θ_{PHT2}, θ_{PHT3}, θ_{PHT4}. The pivots of subgenetic code are used in the RL algorithm for optimal eye tracking control thresholds/parameters, and the RL parameters were determined empirically. Extensive experiments have been carried out using a Pentium 4 with 3 GHz and a Logitech C910 webcam. Eight volunteers participated in the experiments to investigate the effect of diverse image sequences with varying image quality. Face size is restricted to not be varied significantly from frame to frame, and the variance range of detected faces is between 128×128 and 256×256.

6.1 *Data Set*

The image qualities of the test video set is generated as relatively good, moderate, and bad to investigate the learning adaptation capability of the proposed method, where three image categories are defined as three different image contexts. Here, image quality indicates not only the variation of lighting condition, but also that of the eye movement speed and pose. Some examples are given in Fig. 4. Each image sequence has around 2000 image frames, among which half of the image frames are used for GA training, and the remaining frames were used for the test of eye tracking performance.

We created seven videos characterized in terms of the illumination, eye movement speed, and pose movement speed as shown in Table 6, where each video has 1000 frames. In our experiments, the frames with blinking and facial expressions are excluded in the performance evaluation since the aim of our eye tracking is focused on HCI applications where

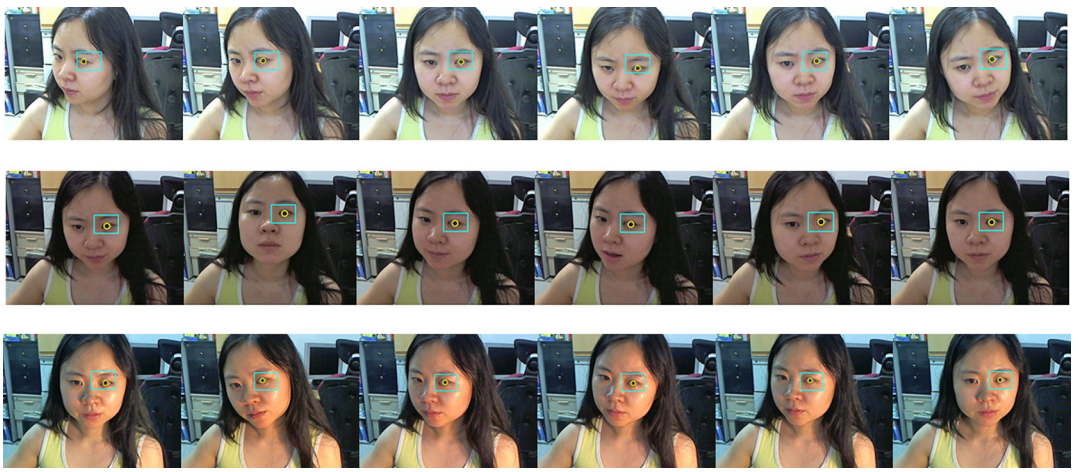


Fig. 5 Illustrations of test videos with varying image quality in terms of illumination, eye movement, and face movement.

Table 6 Characteristics of videos in terms of the illumination, eye movement speed, and pose movement speed.

	Illumination	Eye movement	Pose movement
Video 1	Bad-moderate-good	Moderate	Moderate
Video 2	Bad-bad-bad	Slow	Fast
Video 3	Good-good-good	Fast	Moderate
Video 4	Moderate-bad-good-bad-good	Moderate	Moderate
Video 5	Good-good-good	Fast	Moderate
Video 6	Good-good-good	Slow	Slow
Video 7	Moderate-moderate-moderate	Moderate	Fast

a user's intentional eye movement with a positive attitude can be assumed. Some portions of the test videos are illustrated in Fig. 5.

6.2 *RL and Experimental Parameter Determination*

In this subsection, the effect of the RL parameters and other experimental parameters such as the matching threshold and the real-time matching threshold are analyzed and determined empirically. The RL parameters $\varepsilon \in (0, 1]$ from ε-greedy policy,[16] learning rate $\alpha \in (0, 1]$ and discounting rate $\gamma \in (0, 1]$ from the action-value equation of Q-learning [see Eq. (28)] were investigated. Figure 6 shows average performance of the ε-greedy policy with different values of $\varepsilon = 0.0$, $\varepsilon = 0.1$, and $\varepsilon = 0.15$, where the vertical and horizontal axes are real-time tracking confidence and image frame sequence, respectively. These data are the averages over ten volunteers' videos, where α and γ are set to 0.2 and 0.95, respectively. The parameter ε affects the ratio between exploitation and exploration when a next action is decided. One can notice that $\varepsilon = 0.15$ can achieve faster and better convergence characteristics in real-time eye-tracking confidence.

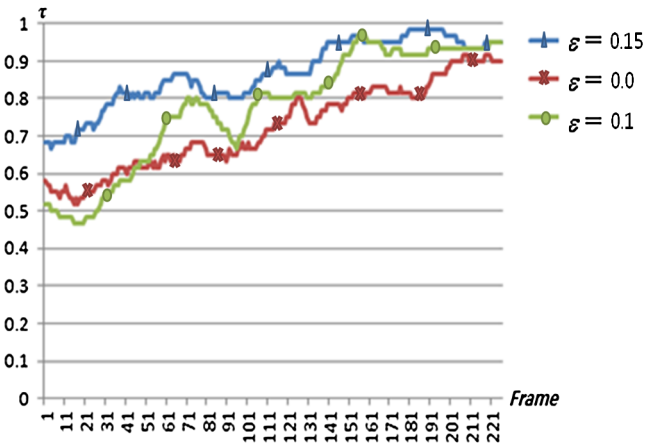


Fig. 6 The effects of greedy policy on the eye tracking performance with different ε values, where the vertical and horizontal axes are real-time tracking confidence and image frame sequence, respectively.

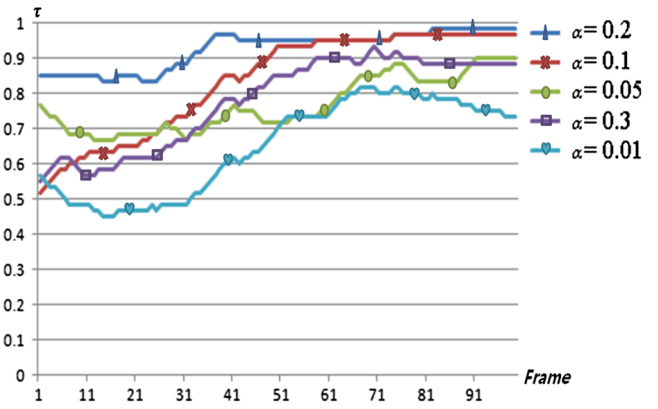


Fig. 7 Effects of RL learning rate on the eye tracking performance with $\alpha = 0.2$, 0.1, 0.05, 0.3, and 0.01.

Figure 7 shows the effect on the eye-tracking performance with different values of $\alpha = 0.2$, 0.1, 0.05, 0.3, and 0.01 by fixing $\varepsilon = 0.15$ and $\gamma = 0.95$. One can notice that $\alpha = 0.2$ gives fast convergence characteristics in real-time tracking confidence.

Figure 8 shows the effects of RL discounting rate with different values, i.e., $\gamma = 0.9$, 0.95, and 0.99 by setting $\varepsilon = 0.15$ and $\alpha = 0.2$.

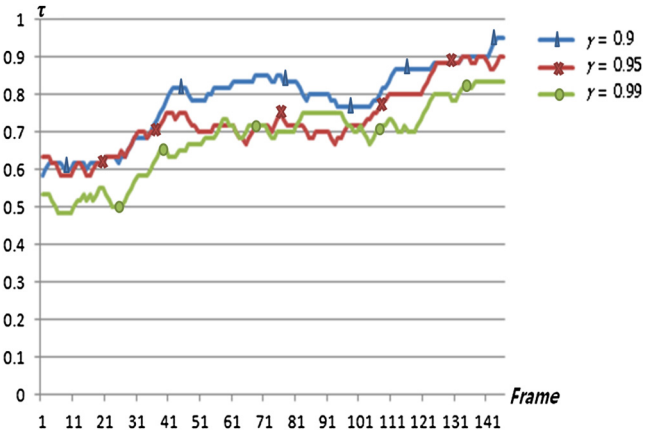


Fig. 8 The effects of RL discount rate on the eye tracking performance with different values of $\gamma = 0.9$, 0.95, and 0.99 by setting $\varepsilon = 0.15$ and $\alpha = 0.2$.

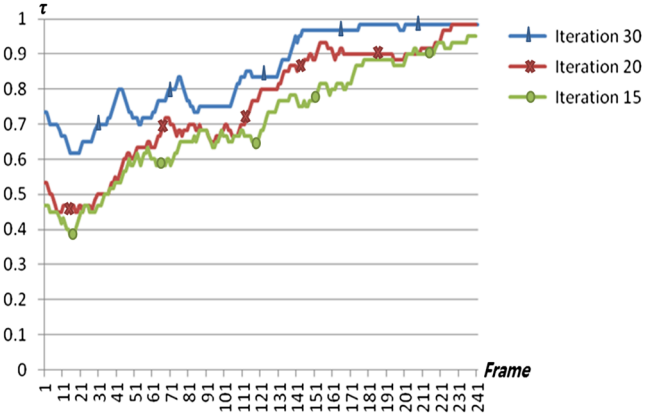


Fig. 9 Real-time tracking confidence with different numbers of iterations at individual image frame ($\varepsilon = 0.15$, $\alpha = 0.2$, $\gamma = 0.95$, and $\delta = 20$).

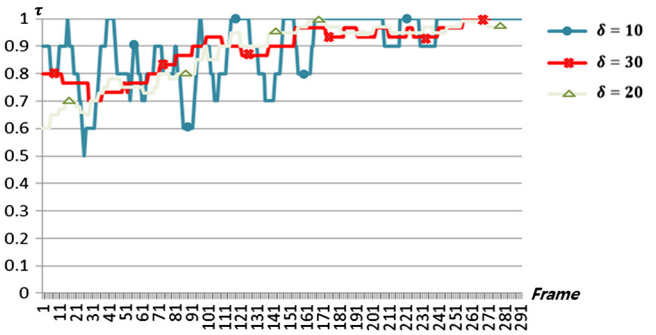


Fig. 10 The effect of different values of the parameter δ in visualization of the real-time tracking confidence ($\varepsilon = 0.15$, $\alpha = 0.2$, and $\gamma = 0.95$).

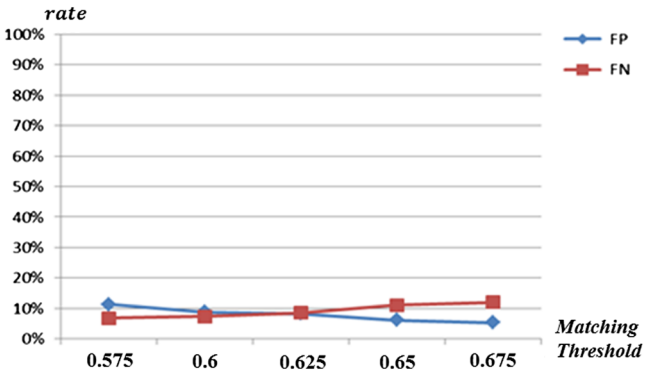


Fig. 11 The effect of matching threshold of real-time tracking confidence. FP and FN indicate the false positive and false negative, respectively.

Considering the real-time constraint, we test the effect of Q-learning step limitation per individual image frame. Figure 9 shows the real-time tracking confidences where the maximum number of Q-steps per frame is restricted to 15, 20, and 30. As the iteration is reduced the tracking performance suffers, but the eye-tracking speed can be increased.

Figure 10 shows the visualization of the real-time tracking confidence with different values of $\delta = 10$, 20, and 30 by setting $\varepsilon = 0.15$, $\alpha = 0.2$, and $\gamma = 0.95$. Since the sufficiently good convergence characteristics of the real-time tracking confidence can be observed starting from $\delta = 20$, we selected $\delta = 20$ considering the time overhead.

Figure 11 shows the rates of the false positive and false negative with different values of real-time matching threshold $t'_m = 0.575$, 0.6, 0.625, 0.65, and 0.675 by setting $\varepsilon = 0.15$, $\alpha = 0.2$, and $\gamma = 0.95$. We decide t'_m as 0.625 since the matching threshold can provide approximately equal values of the false positive and false negative.

The experiments are carried out using the RL parameters of $\varepsilon = 0.15$, $\alpha = 0.2$, and $\gamma = 0.95$, and the experimental parameters $\delta = 20$ in the following discussions.

6.3 *Experimental Results*

Extensive experiments were carried out using various image sequences. The performance was estimated using the seven types of videos gathered from different environments. The EA module decided the pivot of the phenotype control parameters, and the discrete phenotype ranges are determined and used as the RL exploration ranges. While large RL ranges are expected to give high performance, they might require much computation overhead, and the frame processing rate will be decreased, i.e., many input frames are skipped. If the frame rate for eye tracking is reduced too much, the continued variation assumption of illumination might not be valid. The algorithm structures of preprocessing and feature extraction are decided as the binarization and the histogram equalization in the EA module, respectively. The Kalman filter parameters were fixed for the simplicity of analysis. We focused on the adaptation functionalities of the feature extraction threshold, i.e., the binarization threshold and the four partial Hough transform parameters which mainly affect the eye tracking performance. We examined the trade-off the performance and computation overhead by adjusting the maximum number of Q-learning steps per an image frame. Table 7 shows the performance

Table 7 Comparison of the number of successfully tracked frames by non-adaptation, EA-only, and a combination of EA and RL methods with one-step Q-learning.

Image sequence	Frames *A*/*B*	Non-adaptation *C*|fps	EA only *C*|fps	EA + RL(*S*,15) *C*|fps	EA + RL(*S*,20) *C*|fps	EA + RL(*S*,30) *C*|fps
Video 1	825/1000	524|63.4	557|62.5	754|33.0	774|31.8	790|29.4
Video 2	948/1000	734|63.2	789|63.1	895|45.0	916|43.8	937|43.0
Video 3	973/1000	631|63.2	665|62.9	835|27.0	855|24.3	870|19.4
Video 4	976/1000	535|57.4	701|57.2	891|39.7	922|37.1	938|33.6
Video 5	964/1000	631|64.2	702|63.9	778|20.9	788|17.8	827|16.6
Video 6	983/1000	812|54.6	826|53.8	919|39.3	934|37.0	944|35.1
Video 7	992/1000	725|62.0	792|61.7	863|30.0	893|27.3	926|25.0

Note: *A*, *B*, and *C* indicate the numbers of the successful eye area detection, whole test video frames, and successful eye tracking, respectively. The failed eye detection can be calculated by the equation ($B - A$). The number of tracking failures after the successful eye area detection can be calculated by the equation ($A - C$); EA + RL(S, k) means the single-agent Q-learning with maximum k time-steps per an image frame.

comparisons of the proposed method using the single agent Q-learning to the nonadaptation and the adaptation using the EA-only method. In the non-adaptation method, the eye-tracking scheme uses predetermined threshold and control parameters. In the EA-only method, the genetic code decided by training data is used as the threshold and the control parameters. In the combination of EA and RL methods, the RL is applied to decide the control threshold/parameters in the phenotype control space using the pivot values determined by the EA. The real-time computation overhead of the EA-only method is almost equal to the non-adaptation method since the EA evolution is carried out before the eye tracking is performed.

Table 8 shows the trade-off between eye-tracking performance and computation overhead of the proposed multi-agent Q-learning approach by changing the computation time constraints. Note that the multiagent Q-learning can improve eye-tracking performance while increasing computation overhead.

The performance evaluations using precision, recall, harmonic means, and average tracking rate are given in Table 9 for the one-step Q-learning using a single agent and in Table 10 for the multiagent Q-learning, respectively. Table 10 shows that EA + RL with 30 Q-learning steps [EA+ EL$(S, 30)$] achieved the best performance, however, it requires the highest computation overhead. EA + RL$(S, 20)$ achieved moderate performance with moderate computation overhead.

The multiagent Q-learning could achieve better performance than the one-step Q-learning, while adding more

Table 8 Comparison of the number of successfully tracked frames with different computation time constraints using three-agent Q-learning.

Image sequence	Frames *A*/*B*	EA + RL(*M*,15) *C*|fps	EA + RL(*M*,20) *C*|fps	EA + RL(*M*,30) *C*|fps
Video 1	825/1000	810|13.2	813|12.2	817|11.8
Video 2	948/1000	937|18.1	941|16.8	945|17.2
Video 3	973/1000	893|10.8	898|9.3	903|7.8
Video 4	976/1000	951|15.6	953|14.2	956|13.5
Video 5	964/1000	843|8.36	845|6.8	858|6.7
Video 6	983/1000	951|15.7	959|14.5	957|14.0
Video 7	992/1000	942|12.0	952|10.5	957|10.0

Note: *A*, *B*, and *C* indicate the number of the successful eye area detection, that of whole test video frames, and that of successful eye tracking, respectively. The failed eye detection can be calculated by the equation $(B - A)$. The number of tracking failures after the successful eye area detection can be calculated by the equation $(A - C)$; EA + RL(M, k) indicates the multi-agent Q-learning with maximum k time-steps per frame for individual agent. Three agents are employed in this experiment.

Table 9 Evaluation of one-step Q-learning using performance measure precision, recall, harmonic mean, integrated tracking performance, and real-time tracking confidence.

Image sequence	No-adaptation PR|RE|HM|AR	EA –only PR|RE|HM|AR	EA + RL(*S*,15) PR|RE|HM|AR	EA + RL(*S*,20) PR|RE|HM|AR	EA + RL(*S*,30) PR|RE|HM|AR
Video 1	0.89|0.64|0.73|0.63	0.91|0.64|0.75|0.76	0.92|0.85|0.89|0.91	0.93|0.88|0.90|0.93	0.93|0.89|0.91|0.95
Video 2	0.93|0.78|0.85|0.77	0.95|0.79|0.86|0.83	0.93|0.90|0.92|0.94	0.95|0.93|0.93|0.96	0.95|0.94|0.94|0.98
Video 3	0.92|0.67|0.75|0.64	0.94|0.67|0.78|0.68	0.95|0.79|0.86|0.85	0.95|0.84|0.89|0.87	0.96|0.84|0.89|0.89
Video 4	0.73|0.62|0.70|0.61	0.78|0.64|0.70|0.71	0.75|0.70|0.72|0.91	0.78|0.73|0.75|0.94	0.78|0.74|0.75|0.96
Video 5	0.84|0.53|0.65|0.65	0.84|0.55|0.67|0.72	0.79|0.65|0.71|0.80	0.89|0.69|0.78|0.81	0.87|0.72|0.79|0.85
Video 6	0.90|0.79|0.84|0.82	0.93|0.80|0.86|0.84	0.94|0.89|0.91|0.93	0.94|0.91|0.90|0.95	0.94|0.93|0.92|0.96
Video 7	0.92|0.41|0.56|0.73	0.94|0.42|0.58|0.79	0.96|0.76|0.85|0.86	0.97|0.80|0.87|0.90	0.97|0.84|0.90|0.93
Mean	0.87|0.63|0.73|0.69	0.90|0.64|0.74|0.76	0.89|0.79|0.84|0.89	0.92|0.83|0.86|0.91	0.91|0.84|0.87|0.93

Note: Q-learning parameters are set to $\varepsilon = 0.15$, $\alpha = 0.2$, $\gamma = 0.95$, and $\delta = 20$; RL(S, k) means the single agent Q-learning with the maximum iterations constrained by k times; PR indicates the precision, RE the recall, HM the harmonic mean, AR the average tracking rate; EA + RL$(S, 20)$ indicates the combination method using the single agent Q-learning with the constraint of Q-learning steps in each image frame as 20, and so on.

Table 10 Evaluation of the multiagent Q-learning using the performance measures of precision, recall, harmonic mean, integrated tracking performance, and real-time tracking confidence.

Image sequence	EA + RL(M,15) PR|RE|HM|AR	EA + RL(M,20) PR|RE|HM|AR	EA + RL(M,30) PR|RE|HM|AR
Video 1	0.93|0.90|0.91|0.98	0.93|0.91|0.92|0.98	0.93|0.92|0.92|0.99
Video 2	0.94|0.94|0.94|0.99	0.94|0.94|0.94|0.99	0.95|0.94|0.95|0.99
Video 3	0.95|0.86|0.90|0.92	0.95|0.87|0.91|0.92	0.96|0.88|0.92|0.92
Video 4	0.77|0.75|0.76|0.97	0.77|0.75|0.76|0.97	0.78|0.76|0.77|0.97
Video 5	0.87|0.73|0.80|0.87	0.88|0.75|0.81|0.87	0.88|0.76|0.81|0.89
Video 6	0.94|0.91|0.93|0.96	0.94|0.92|0.93|0.97	0.94|0.92|0.93|0.97
Video 7	0.96|0.86|0.90|0.94	0.96|0.80|0.91|0.95	0.97|0.89|0.92|0.96
Mean	0.90|0.85|0.88|0.95	0.91|0.85|0.88|0.95	0.92|0.87|0.89|0.96

Q-learning parameters are set to $\varepsilon = 0.15$, $\alpha = 0.2$, $\gamma = 0.95$, and $\delta = 20$; EA + RL(M, k) means the multiple agents Q-learning with maximum iterations constrained by k times. Three agents are employed in this experiment.

computation overhead. In Table 10, the three-agent Q-learning with different time constrains are investigated where the total number of Q-learning steps per an image frame of each agent is restricted from 15, 20, and 30.

The proposed method can guarantee optimal real-time performance by compromising the trade-off between the performance and the computation overhead. The tradeoffs of execution speed and tracking performance among different system architectures are compared in Fig. 12 using the average tracking rate, precision, recall, and harmonic mean, respectively. The horizontal axes indicate the time overhead using fps. Q-learning parameters are set to $\varepsilon = 0.15$, $\alpha = 0.2$, $\gamma = 0.95$, and $\delta = 20$. Notice that higher performance can be obtained if the time overhead is increased.

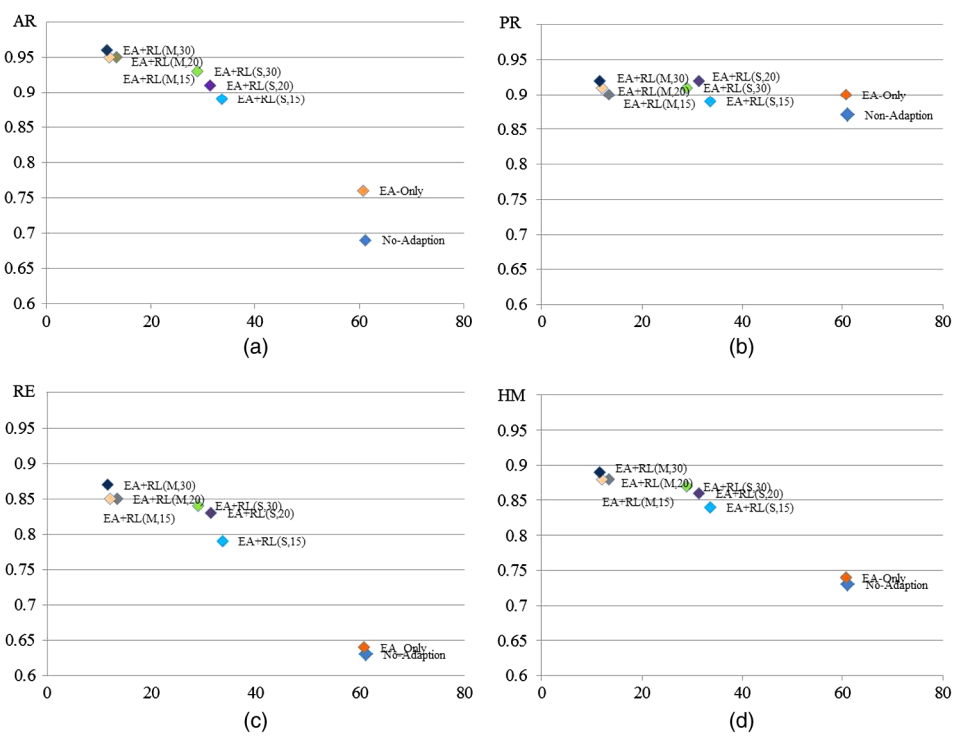


Fig. 12 The tradeoffs of the execution speed and the tracking performance among different system architectures where Q-learning parameters are set to $\varepsilon = 0.15$, $\alpha = 0.2$, $\gamma = 0.95$, and $\delta = 20$: (a) the average of real-time eye tracking, (b) the precision, (c) the recall, and (d) the harmonic mean. The horizontal axes indicate the time overhead using fps and AR indicates the average tracking rate, PR the precision, RE the recall, and HM the harmonic mean.

7 Concluding Remarks

Combining the EA approach and the RL algorithm, i.e., the difficulty of the EA in a real-time adaptation and the curse of huge search space of the RL algorithm, the proposed method can manage to achieve the performance of the eye-tracking scheme in terms of accuracy and speed. The state-of-the-art of eye-tracking techniques relies on either high cost and high quality cameras/equipment or a strongly controlled situation. To overcome limitations, possible configurations of the eye-tracking scheme are explored in the genotype and phenotype control spaces by the EA and the RL, respectively. The EA is responsible for exploring the genotype control space using a collected training image set, and the RL algorithm organizes an evolved genotype into a reactive phenotype that optimizes the eye-tracking scheme in real-time performance. The EA encodes and stores the learned optimal scheme in terms of a genetic code which denotes an eye-tracking algorithm structure and associated thresholds/parameters. The RL algorithm defines its internal environmental states in a phenotype control space and mainly performs interactive learning and adaptation in real-time.

The main contribution of the proposed method compared to other popular adaptive systems is to provide a design technique that can manage an intrinsic uncertainty of vision based HCI into a tractable computation space for controlling the algorithm structure and associated thresholds/parameters. It can optimize the performance of low-cost eye-tracking schemes in accordance with a changing environment. The proposed method defines the internal RL environment connecting with the genotype control space instead of the input image directly as other RL environment[21] so that the interactive characteristics of the RL algorithm are fully utilized by exploring the threshold/parameter space influenced by a changing external environment. Since the main focus of the proposed method is to optimize an eye-tracking system with acceptable computation resources, we discussed using a relatively simple eye-tracking method. However, the proposed framework can be extended to other eye tracking methods encountering difficulties in achieving optimal performance due to unstable operational environments. The future direction of our research is to apply the proposed method in a challenging vision-based HCI application for next-generation computing such as eye cursors, eye keyboards, eye mice, and vision-based emotion detection.

Acknowledgments

This work was supported by an Inha University research grant. And following are results of a study on the "Leaders in Industry-university Cooperation" Project, supported by the Ministry of Education, Science & Technology and Inha University.

References

1. D. W. Hansen et al., "In the eye of the beholder: a survey of models for eyes and gaze," *IEEE Trans. Pattern Anal. Mach. Intell.* **32**(3), 478–500 (2010).
2. A. T. Duchowski, "A breadth-first survey of eye-tracking applications," *Behav. Res. Methods Instrum. Comput.* **34**(4), 455–470 (2002).
3. A. Sein et al., "Eyeing a real-time human-computer interface to assist those with motor disabilities," *IEEE Potentials* **27**(3), 19–25 (2008).
4. C. A. Hennessey and P. D. Lawrence, "Improving the accuracy and reliability of remote system-calibration-free eye-gaze tracking," *IEEE Trans. Biomed. Eng.* **56**(7), 1891–1900 (2009).
5. M. Nabati and A. Behrad, "Camera mouse implementation using 3D head pose estimation by monocular video camera and 2D to 3D point and line correspondences," in *Proc. IEEE 5th Int. Symposium on Telecommunications*, pp. 825–830, IEEE, Tehran, Iran (2010).
6. L. J. G. Vazquez, M. A. Minor, and A. J. H. Sossa, "Low-cost human computer interface voluntary eye movement as communication system for disabled people with limited movements," in *Proc. IEEE Pan American Health Care Exchanges*, pp. 165–170, IEEE, Rio de Janeiro (2011).
7. J. S. Agustin et al., "Low-cost gaze interaction: ready to deliver the promises," in *Proc. CHI'09 Extended Abstracts on Human Factors in Computing Systems*, pp. 4453–4458, ACM, New York (2009).
8. B. Bhanu and J. Peng, "Adaptive integrated image segmentation and object recognition," *IEEE Trans. Syst. Man Cybern. C Appl. Rev.* **30**(4), 427–441 (2000).
9. D. P. Muni, N. R. Pal, and J. Das, "A novel approach to design classifiers using genetic programming," *IEEE Trans. Evol. Comput.* **8**(2), 183–196 (2004).
10. J. H. Holland, "Genetic algorithms," *Sci. Am.* **267**(1), 66–72 (1992).
11. J. Clune et al., "Evolving coordinated quadruped gaits with the HyperNEAT generative encoding," in *Proc. IEEE Congress on Evolutionary Computation*, pp. 2764–2771, IEEE, Trondheim, Norway (2009).
12. C. J. C. H. Watkins, "Learning from delayed rewards," PhD Thesis, King's College (1989).
13. K. Yanai and H. Iba, "Multi-agent robot learning by means of genetic programming: solving an escape problem," in *Proc. 4th Int. Conf. Evolvable Syst. From Biology to Hardware*, Lecture Notes in Computer Science, Vol. 2210, pp. 192–203, Springer-Verlag, Berlin, Heidelberg (2001).
14. S. Kamio and H. Iba, "Adaptation technique for integrating genetic programming and reinforcement learning for real robots," *IEEE Trans. Evol. Comput.* **9**(3), 318–333 (2005).
15. M. A. Wiering and H. V. Hasselt, "Ensemble algorithms in reinforcement learning," *IEEE Trans. Syst. Man Cybern. B Cybern.* **38**(4), 930–936 (2008).
16. R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, MIT Press, Cambridge, Massachusetts (1998).
17. N. Mastronarde and M. V. D. Schaar, "Online reinforcement learning for dynamic multimedia systems," *IEEE Trans. Image Process.* **19**(2), 290–305 (2010).
18. J. Valasek et al., "Improved adaptive reinforcement learning control for morphing unmanned air vehicles," *IEEE Trans. Syst. Man Cybern. B Cybern.* **38**(4), 1014–1020 (2008).
19. H. Sellahewa and S. A. Jassim, "Image-quality-based adaptive face recognition," *IEEE Trans. Instrum. Meas.* **59**(4), 805–813 (2010).
20. E. J. Koh, M. Y. Nam, and P. K. Rhee, "A context-driven Bayesian classification method for eye location," in *Proc. 8th Int. Conf. Adaptive and Natural Computing Algorithms*, Lecture Notes in Computer Science, Vol. 4432, pp. 517–524, Springer-Verlag, Berlin, Heidelberg (2007).
21. J. Peng and B. Bhanu, "Delayed reinforcement learning for adaptive image segmentation and feature extraction," *IEEE Trans. Syst. Man Cybern. C Appl. Rev.* **28**(3), 482–488 (1998).
22. P. K. Rhee, M. Y. Nam, and L. Wang, "Pupil location and movement measurement for efficient emotional sensibility analysis," in *Proc. IEEE Int. Symp. on Signal Process. and Inform. Technol.*, pp. 1–6, IEEE, Luxor (2011).
23. J. Cauchie et al., "Optimization of an Hough transform algorithm for the search of a center," *Pattern Recognit.* **41**(2), 567–574 (2008).
24. M. S. Arulampalam et al., "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," *IEEE Trans. Signal Process.* **50**(2), 174–188 (2002).
25. D. J. Jobson, Z. Rahman, and G. A. Woodell, "A multi-scale retinex for bridging the gap between color images and the human observation of scenes," *IEEE Trans. Image Process.* **6**(7), 965–976 (1997).
26. M. Y. Nam and P. K. Rhee, "An efficient face recognition for variant illumination condition," in *Proc. IEEE Int. Symp. on Intell. Signal Process. and Commun. Syst.*, pp. 111–115, IEEE, Incheon, South Korea (2005).
27. J. A. Canny, "Computational approach to edge detection," *IEEE Trans. Pattern Anal. Mach. Intell.* **PAMI-8**(6), 679–698 (1986).
28. R. E. Kalman, "A new approach to linear filtering and prediction problems," *J. Basic Eng.* **82**(1), 35–45 (1960).
29. G. Welch and G. Bishop, "An Introduction to the Kalman Filter," UNC-Chapel Hill, TR95-041 (2000), http://www.cs.unc.edu.
30. Q. Chen et al., "Two-stage object tracking method based on kernel and active contour," *IEEE Trans. Circuits Syst. Video Technol.* **20**(4) 605–609 (2010).
31. M. R. Rajamani and J. B. Rawlings, "Estimation of the disturbance structure from data using semidefinite programming and optimal weighting," *Automatica* **45**(1), 142–148 (2009).
32. M. S. Grewal and A. P. Andrews, *Kalman Filtering Theory and Practice*, Prentice Hall, Upper Saddle River, New Jersey (1992).
33. M. Abdel-Mottaleb and M. H. Mahoor, "Application notes—algorithms for assessing the quality of facial images," *IEEE Comput. Intell. Mag.* **2**(2), 10–17 (2007).

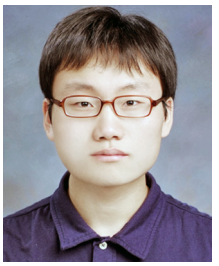
34. Q. Li and Z. Wang, "Reduced-reference image quality assessment using divisive normalization-based image representation," *IEEE J. Sel. Topics Signal Process.* **3**(2), 202–211 (2009).
35. S. Ross et al., "A Bayesian approach for learning and planning in partially observable Markov decision processes," *J. Mach. Learn. Res.* **12**(5), 1729–1770 (2011).
36. L. Busoniu et al., "A comprehensive survey of multiagent reinforcement learning," *IEEE Trans. Syst. Man Cybern. C Appl. Rev.* **38**(2), 156–172 (2008).
37. S. Singh, "Intrinsically motivated reinforcement learning: an evolutionary perspective," *IEEE Trans. Autom. Mental Devel.* **2**(2), 70–82 (2010).
38. L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: a survey," *J. Artif. Intell. Res.* **4**(1), 237–285 (1996).
39. C. Wolf and J.-M. Jolion, "Object count/area graphs for the evaluation of object detection and segmentation algorithms," *Int. J. Doc. Anal.* **8**(4), 280–296 (2006).
40. Z. Kadal et al., "Tracking-learning-detection," *IEEE Trans. Pattern Anal. Mach. Intell.* **34**(7), 1409–1422 (2012).
41. T. Fawcett, *ROC Graphs: Notes and practical Considerations for Researchers*, Kluwer Academic Publishers, The Netherlands (2004).

Yan Shen received her BS degree in computer science from Inha University, Incheon, Republic of Korea, in 2011. She is currently pursuing the master's degree at Inha University, where she is majoring in computer science and engineering. Her research interests include recommender systems, computer vision, intelligent computers, and *HCI*.

Hak Chul Shin received his BS degree in computer science from the Inha University, Incheon, Republic of Korea, in 2010. He is currently pursuing the master's degree at Inha University where he is majoring in computer science and engineering. His research interests include imaging processing, object detection, and trajectory analysis.

Won Jun Sung received his BS degree in computer science from the Kongju University, Cheonan, Republic of Korea, in 2012. He is currently pursuing the master's degree at Inha University, where he is majoring in computer science and engineering. His research interests include imaging processing, object detection, and face recognition.

Sarang Khim is currently pursing his BS degree in computer and information engineering from Inha University, Incheon, Republic of Korea, in 2012. His research interests include pattern computer vision, intelligent technology, and artificial intelligence.



Honglak Kim received his BS degree in computer and information engineering from Inha University, Incheon, Republic of Korea, in 2012. His research interests include pattern recognition, machine learning, and computer vision.

Phill Kyu Rhee received his BS degree in electrical engineering from the Seoul University, Seoul, Republic of Korea, his MS degree in computer science from the East Texas State University, Commerce, Texas, and his PhD degree in computer science from the University of Louisiana, Lafayette, Louisiana, in 1982, 1986, and 1990, respectively. During 1982 to 1985, he was working in the System Engineering Research Institute, Seoul, Republic of Korea, as a research scientist. In 1991, he joined the Electronic and Telecommunication Research Institute, Seoul, Republic of Korea, as a senior research staff member. Since 1992, he has been an associate professor in the Department of Computer Science and Engineering of the Inha University, Incheon, Republic of Korea, and since 2001, he is a professor in the same department and university. His current research interests are pattern recognition, machine intelligence, and autonomic cloud computing. He is a member of the IEEE Computer Society and Korea Information Science Society.