Estudio de comportamiento ante ruido de un controlador evolutivo soportado por una red neuronal hardware de diseño específico

Mg. Marcelo A. Tosini

Facultad de ciencias Exactas
Universidad Nacional del Centro de la Provincia de Buenos Aires
Tandil (7000) – Buenos Aires – Argentina
Tel/Fax: (+54) 2293 439680
Email: mtosini@exa.unicen.edu.ar

Resumen.

La implantación de controladores neuronales evolucionados genéticamente en dispositivos de hardware reales requiere prestar especial atención a las perturbaciones externas que influyen la percepción y movimientos del controlador.

Se realiza un análisis de la influencia de distintos tipos de perturbaciones en controladores evolucionados con diferentes parámetros estructurales o temporales y su efecto sobre el rendimiento final.

Los experimentos se realizan sobre controladores evolucionados para la resolución de un problema clásico de discriminación de objetos. Dichos controladores son sometidos a la influencia de ruidos en sus entradas y salidas, entrenados con distintos grados de reacción y probados en ambientes con perturbaciones en los objetos a discriminar.

Se pretende, de este modo, recabar información sobre el comportamiento de los agentes genéticamente evolucionados ante distintos eventos que pueden presentarse en un escenario real. Esta información puede ser relevante para el diseño de controladores hardware sometidos a ruidos de sensores, actuadores o interferencias internas al propio controlador.

1. Introducción

El campo de la robótica evolutiva se orienta en dos campos o áreas bien delimitados sobre todo en los últimos años. Por un lado hay trabajos orientados al aspecto teórico de la disciplina, dejando más de lado el punto de vista de la ingeniería [1]. Este enfoque dio buenos resultados reflejados sobre todo en estudios teóricos acerca de complejidad de las tareas realizadas y en el ámbito de la aproximación simbólica a los problemas [2-3].

Otros trabajos aplican más al estudio de problemas referentes a controladores evolucionados genéticamente soportados por dispositivos implementados en hardware. En este ámbito hay que tener en cuenta factores colaterales relacionados con aspectos tales como interferencias eléctricas (o de otra índole) en los sensores, en

los actuadores (motores o circuitos de control de los mismos) o del propio ambiente (diferencias entre el comportamiento del ambiente real y el modelo simulado) [4-5].

Al momento del diseño del controlador neuronal los diferentes ruidos se incluyen en las ecuaciones que modelan la red neuronal de control en tres puntos diferentes: 1) agregado a los valores de los sensores (ruido aditivo); 2) como coeficiente de fricción de los motores, ruido mecánico o ruido eléctrico amplificado por los controladores de los motores) ó 3) como ruido propio al modelo de cada neurona (precisión limitada).

El ruido ambiente puede ser de dos tipos: Característico del mundo real que no puede modelarse correctamente en el entorno simulado; y ruido propio eventual del escenario real. En este último caso el modelo debe ser capaz de generar situaciones en las que no existe perturbación y otras en las que la perturbación puede aparecer eventualmente en algún momento del experimento. En general [6], para capturar todas las características de un problema en un controlador es necesario considerar todos los aspectos subyacentes – ruidos, incertezas- puesto que la simplificación del modelo al excluir alguno de ellos aumenta el riesgo de obtener controladores buenos pero más frágiles.

En experimentos reportados por Gallardo [7] para el problema de la incertidumbre en el comportamiento de sensores y actuadores de controladores neuronales aplicados a agentes en laberintos, los resultados muestran que las soluciones obtenidas en ambientes con ruido son menos óptimas que aquellas sin ruidos ya que los controladores se ven forzados a alejarse más de los obstáculos a fin de esquivarlos con seguridad.

En todos los casos el ruido está presente en el nivel inferior (nivel neuronal), mientras que la manera en que las tareas son realizadas se ve a alto nivel (nivel de comportamiento). Esto indica que siempre es necesario considerar los efectos a nivel neuronal a fin de entender su influencia sobre el nivel comportamental.

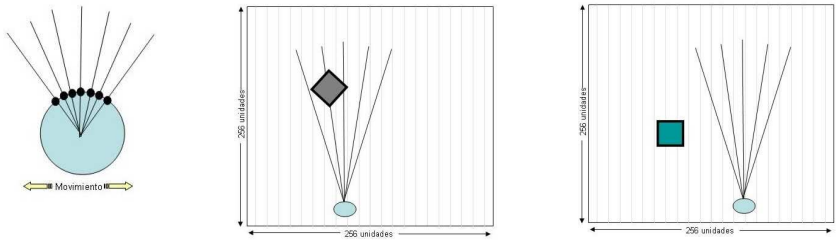
La estructura del trabajo es la siguiente: La sección 2 describe el modelo usado para las pruebas, basado en experimentos sugeridos por Beer en [8] y la arquitectura hardware de destino. En la sección 3 se muestran los experimentos realizados sometiendo el modelo general a la influencia de distintas interferencias en entradas y salidas. En las siguientes secciones se analizan los resultados obtenidos y se realiza una discusión de los mismos comparando las distintas pruebas realizadas a fin de obtener conclusiones que permitan su aplicación futura en ambientes reales.

2. Modelo evolutivo y ambiente de simulación

Modelo Utilizado

Se utiliza como modelo para este trabajo una tarea clásica de discriminación entre objetos por parte de un agente genéticamente evolucionado con capacidad de movimiento horizontal y situado en la base de un escenario de 256 por 256 unidades., Dicho agente se evoluciona para detectar objetos cayendo verticalmente desde la parte superior del escenario (Figura 1). Su función es la de capturar objetos de un tipo (rombos) y evadir los de un segundo tipo (cuadrados).

El agente tiene un cuerpo circular de 32 unidades de diámetro con 7 sensores frontales de distancia distribuidos sobre un ángulo visual de π/6. La intersección entre el haz de un sensor y un objeto causa la inyección en la correspondiente neurona sensora de un valor inversamente proporcional a la distancia entre dicho objeto y el sensor. El valor inyectado es 10 para objetos a distancia cero del agente y 0 para objetos situados a la máxima distancia (256 unidades).



a) b) b)

Fig. 1. Ambiente de simulación. a) Un agente con 7 sensores frontales de distancia y capacidad de desplazamiento horizontal. b) Tarea de captura de objetos tipo rombo. c) evasión de objetos tipo cuadrado.

El modelo neuronal del controlador posee 7 neuronas de entrada (una por sensor) y 2 neuronas motoras (el sentido de desplazamiento se determina por la diferencia entre sus valores de salida). Ambos niveles se vinculan a través de una capa oculta de 5 neuronas totalmente interconectadas (Figura 2). El modelo completo responde a una red neuronal recurrente de tiempo continuo tipo CTRNN (continuous Time Recurrent Neural Network) [9] con las siguientes ecuaciones de estado para las neuronas de entrada, ocultas y motoras respectivamente:

$$\frac{\partial y}{\partial t} i = \frac{1}{\tau_i}\left(-y_i + (I_i + rs_i)\right) \tag{1}$$

$$\frac{\partial y}{\partial t} i = \frac{1}{\tau_i}[-y_i + \sum_{j=1}^{7} w_{ij}\sigma(g_j(y_j + \theta_j)) + rn_i] \tag{2}$$

$$y_i = \sigma(g_j(y_j + \theta_j)) \tag{3}$$

siendo y_i, el estado de la i-esima neurona; τ_i la constante de tiempo; w_{ij}, la magnitud de la conexión de la neurona j a la neurona i; $\sigma(x)$, la función de activación logística estándar:

$$\sigma(x) = \frac{1}{1 + e^{-g(x+\theta)}} \tag{4}$$

con bias θ; ganancia g; I_i las magnitudes de las 7 entradas de sensores; y rs_i y rn_i

valores de ruido aleatorio de entrada y neuronal respectivamente (usados en los diferentes experimentos).

En todos los casos se usa el método de integración de Euler con pasos de integración dependientes de cada experimento y una constante de tiempo τ_i igual a 1.0.

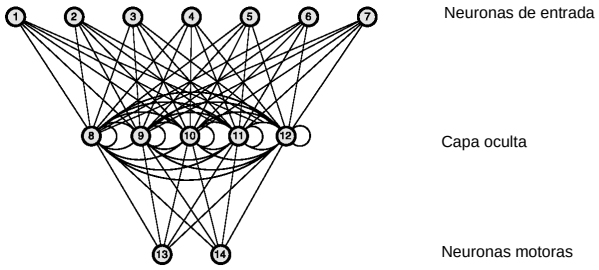


Fig. 2. Modelo neuronal de los controladores. Consiste de una capa de entrada de siete neuronas conectadas a los sensores, una capa interna de cinco neuronas completamente interconectadas y dos neuronas de salida que proveen las magnitudes de desplazamiento a izquierda o derecha del agente.

Los objetos caen a velocidad constante desde posiciones iniciales *(x,y)* con *x* en el rango [-64,64] e *y* en el rango [192,256]. La posición del agente queda determinada por la ecuación:

$$x\_agente_{i+1} = x\_agente_i + (S_1 - S_2) \times vel\_agente \times rm_i \times h \qquad (5)$$

Siendo *x_agente* la posición horizontal del agente; S_1 y S_2 las salidas de las neuronas motoras; *h* el paso de integración y rm_i el ruido aleatorio de la salida en el tiempo *i*.

Ambiente de simulación

El modelo del controlador se probó en una red neuronal emulada materializada por una ruta de datos sistólica (figura 4) compuesta por diferentes bloques aritméticos orientados a las distintas operaciones básicas matriciales (multiplicación matriz-vector, suma de vectores, producto externo de vectores) así como a operaciones específicas del procesamiento neuronal tal como el cálculo de la función de salida y la derivada de las salidas de cada neurona [10].

Los bloques sistólicos (formados por procesadores elementales) son de longitud variable adecuada a la estructura de la red neuronal que se esta procesando (7, 5, 2 neuronas)

Los datos de trabajo se almacenan en estructuras de datos tipo FIFO. Las matrices de pesos sinápticos se almacenan en varias FIFOs organizadas por filas. Esto es, la *n-esima* FIFO de pesos almacena la fila *n* de todas las matrices de pesos. La aritmética de trabajo es acotada a 12 bits.

La FIFO de datos almacena tanto las salidas de cada etapa como los valores de

entrada de la red. Esto facilita la disponibilidad de todos los vectores de datos a la hora de realizar los cálculos del fragmento de código (5).

Los procesadores elementales (PE) usados implementan básicamente unidades de multiplicación y suma de valores con pequeños cambios entre los tres tipos usados [11][12][13]

La función de activación sigmoidea se implementa como un subcircuito combinacional que materializa funciones polinomiales aproximadas a determinadas potencias de 2.

Particularmente se desarrolló un circuito para la función de activación cuya entrada es un valor entero de 16 bits [-32768..32767] y su salida es un entero sin signo de 16 bits [0..65535]. Los valores complementados de la entrada [-32768..0] darán los correspondientes valores de salida en el rango [0..32768].

Entrada	Salida
0..2047	32768..40704
2048..4095	40705..48896
4096..8191	48897..57088
8192..12287	57089..61184
12288..16383	61185..63232
16384..32767	63233..65535

Tabla 1. Funciones polinomiales aproximadas a potencias de 2

En cada experimento el controlador se evolucionó en software y modelo del agente evolucionado se cargó en el emulador del hardware de la red neuronal modelada en un Lenguaje de descripción de hardware (VHDL) con procesos adicionales que simulaban el ambiente externo al agente (escenario y objeto a reconocer).

3. Experimentos realizados

Los experimentos se agruparon en tres categorías para evaluar el comportamiento del controlador con distintas influencias de ruido (Tabla 1).

En todos los experimentos se evolucionó a los controladores durante 2048 generaciones del algoritmo genético con una población de 200 individuos, cada uno sometido a 3 pruebas a fin de obtener un rendimiento promedio. Dicho rendimiento refleja en porcentaje la distancia horizontal del agente cuando los objetos (un rombo o un cuadrado) llegan a la altura del mismo. De este modo, el mejor rendimiento corresponderá a un agente que se aproxime a distancia 0 (mínima) de un rombo y se aleje a distancia máxima de un cuadrado.

Para todos los casos el algoritmo genético generó individuos con las siguientes características: genotipo de longitud 47 (pesos, bias y ganancias de la CTRNN) en el rango [-5; 5], probabilidad de cruzamiento de 0.25, selección de 2 individuos para cruzamiento y mutación proporcional con desviación de 0.0625 (valores adecuados a su representación en base 2; de modo que, por ejemplo, $h=0.015625_{(10}$ es $0.000001_{(2}$)

igual a 1 * 2^{-6} y equivalente a un desplazamiento a la derecha en 6 lugares del resultado.

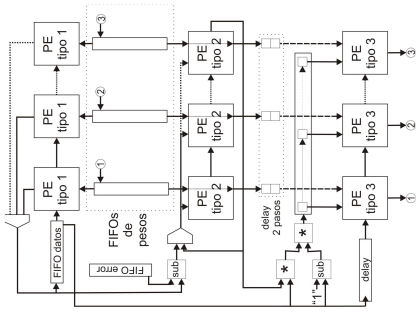


Fig. 3. Ruta de datos del procesador neuronal con las etapas de prueba y de *backpropagation*.

	Ruido	Experimentos	Descripción
1	Neuronal	r_n = 0.0625, 0.1875 ,0.625 y 1.0	• Reactivo con h = 0.015625 • Conexiones intre neuronas
		r_n = 0.0625 y 1.0	• Reactivo con h = 1.0
2	Entrada/ salida	r_s = [0;3.0] r_m = [0;3.0]	• Controlador no reactivo • h = 0.015625 • Ruidos de entrada/salida aleatorio en rango…
		r_s = [0;10.0] r_m = [0;10.0]	
3	Ambiente	r_a = [-1.0;1.0]	• Reactivo con h = 0.25
		r_a = [-1.0;1.0]	• Reactivo con h = 1.0

Tabla 2. Tres grupos experimentos orientados a probar el comportamiento del modelo ante la influencia de ruidos neuronales, de entrada/salida o en el ambiente. Los valores r_n, r_s y r_m son los ruidos incluidos en las ecuaciones 1, 2 y 5 respectivamente; el valor r_a representa el nivel de ruido sobre la posición horizontal de los objetos cuando caen. El parámetro h representa el paso de integración.

En la segunda categoría se evolucionaron controladores con rangos de ruidos aleatorios en sensores y motores. Los primeros, con ruidos en el rango [0; 3.0] representan perturbaciones de 30% en la función normal del componente comprometido. Los segundos, con interferencias aleatorias de hasta 100%, modelan perturbaciones severas en la funcionalidad de los componentes. Particularmente, el controlador puede recibir información confusa que perturbe tanto la forma como la altura a la que se encuentra el objeto, y de este modo, es de suponer que confunda ambos tipos de objetos.

El tercer grupo de experimentos modela el caso de perturbaciones en el ambiente que pueden influir en las decisiones que realiza el agente en su tarea de capturar rombos y evadir cuadrados.

Para probar el comportamiento de los controladores se realizaron dos tipos de experimentos. a) evolución de controladores *no reactivos* y *reactivos* con objetos que alteran su desplazamiento horizontal en algún momento aleatorio de su caída; y b) desplazamiento horizontal zigzagueante de los objetos a partir de un punto aleatorio

de su caída, o sea, el objeto cae con un desplazamiento *x* determinado y fijo hasta la mitad de su altura original. A partir de ese momento su posición *x* comienza a variar aleatoriamente en un rango [-1;1].

La hipótesis de trabajo para los controladores evolucionados en estos ambientes es que los mismos pierden capacidad de predicción de la posición en la que estarán los objetos y, por ende, su rendimiento decaerá.

4. Resultados obtenidos

En la etapa de prueba se sometió a los agentes a 41 objetos de cada tipo (rombos y cuadrados) en trayectoria descendente desde posiciones horizontales en el rango [-20;20], puesto que los agentes siempre comienzan su búsqueda en el centro del escenario (desplazamiento 0 horizontal).

La figura 4 muestra los rendimientos del primer grupo de experimentos. En todos los casos, los controladores *no reactivos* mostraron rendimientos superiores al 99%, mientras que los *reactivos*, aún para niveles de ruido no tan altos como en el caso anterior, tuvieron peor desempeño. De hecho, en el caso de niveles de ruido en rango [0;0.1] el rendimiento baja a 93%.

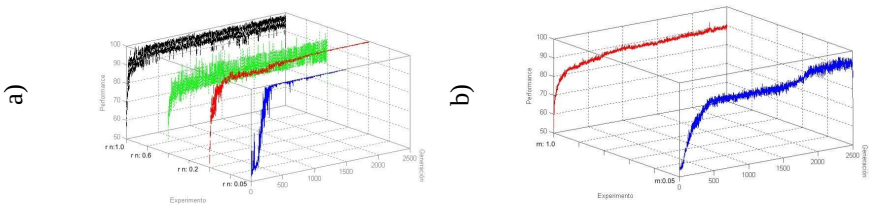


Fig. 4. Rendimientos de los controladores ante ruido neuronal. a) No reactivos con ruidos neuronales (r_n) de 0.0625; 0.1875; 0.625 y 1.0. b) controladores reactivos con ruidos neuronales de 0.0625 y 0.125.

La capacidad de discriminación de los controladores de este primer grupo se observa en la figura 5. En 5-a) y 5-b) se muestra el desempeño de los 2 controladores extremos con ruidos neuronales de 0.05 y 1.0 respectivamente. En el primer caso el controlador puede capturar todos los objetos con un comportamiento similar de búsqueda para cada uno. Al aumentar el ruido neuronal el controlador pierde capacidad de discriminación, como se nota en la figura 5-b), con un comportamiento diferente para objetos cayendo de distintas posiciones y eventos de "no captura" de algunos de ellos.

En figura 5-c) y 5-d) se observa el comportamiento de los controladores reactivos.

En ambos casos la capacidad de captura de los objetos se mantiene aunque el comportamiento del controlador es más errático. Esto último se debe a que los agentes son reactivos puros y, por ende, carecen de memoria temporal que amortigüe su movimiento.

En el segundo grupo de experimentos se evolucionó y probó a los controladores con ruidos de entrada y salida variando aleatoriamente en rangos de [0; 3] y [0; 10] a fin de poder comparar controladores puros (sin interferencias de entrada y salida)

sometidos a niveles de ruido y controladores entrenados con ruido de entrada/salida testeados en entornos “sin” y “con” interferencia.

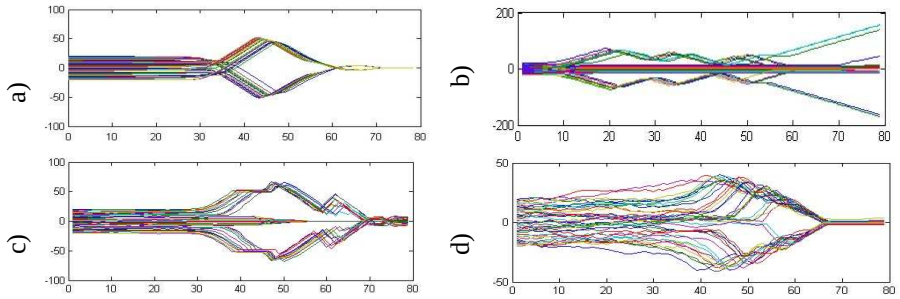


Fig. 5. Tarea de discriminación de objetos para 41 rombos en controladores no reactivos y reactivos sometidos a ruido neuronal. El eje *x* representa el tiempo y el eje *y* la distancia horizontal de los objetos respecto del agente. a) y b) controladores no reactivos con diferentes valores de h; c) y d) controladores reactivos.

5. Discusión

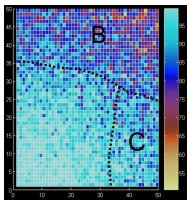
A fin de analizar la influencia de los dos ruidos (entrada y salida) en los controladores obtenidos, se los sometió a un test integrado con los ruidos aleatorios de entrada y de salida desde 0 a 5.0 con incrementos de 0.1 (ó 0.2, según el caso). Para cada uno se calculó el rendimiento promedio de 100 controladores evaluando 6 objetos (3 rombos y 3 cuadrados) cada controlador.

Los resultados se pueden ver en la figura 6, con un controlador entrenado con r_s = 3.0 y r_m = 3.0 y probado con niveles de ruido en sensores y motores en rango [0; 10] (figura 6-a) y en rango [0; 5] (figura 6-b); y un controlador entrenado con ruidos en rango [0; 10] y probado en el mismo rango.

El controlador de la figura 6-b es robusto en casi todo el espectro de ruidos a los que se lo sometió con rendimientos superiores al 97% en la zona A y mayoritariamente superiores al 93% para todos los rangos de ruido testeados. También se observa que con niveles de ruido mayores a 5 el rendimiento cae sensiblemente (zona B). No ocurre lo mismo con valores de ruido alto en los motores donde el rendimiento se encuentra mayormente entre 90 y 98% (zona C).

Los resultados de los experimentos de ruido del ambiente se discuten a continuación. La figura 7 muestra el comportamiento de los agentes para los 2 tipos de experimentos realizados. Sometidos a objetos que varían drásticamente su posición ninguno de los controladores (figuras 7-a y 7-b) logró un rendimiento aceptable -68% para *no reactivos* y 65% para *reactivos*-.

Los controladores entrenados y testeados con objetos de trayectoria ondulante presentaron un comportamiento diferente a los del caso anterior. De hecho, si bien el rendimiento del controlador *no reactivo* (72%) no fue significativamente superior a su antecesor, si mejoró la correspondiente al controlador *reactivo* que fue de 92%. Esto se observa en las figuras 7-c) y 7-d).

a) 

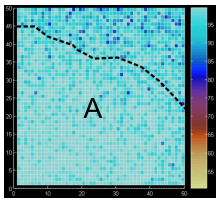
b) 

Fig. 6. Rendimientos medidos para controladores sometidos a ruidos de entrada y salida desde 0 hasta 5.0 con incrementos de 0.1 ó 0.2. El eje x representa el nivel de ruido de salida y el eje y el nivel de ruido de entrada. Cada punto de las gráficas representa el rendimiento obtenido como promedio de 100 pruebas de cada controlador con 6 objetos en cada una. a) controlador entrenado en rango de ruido [0; 3] y probado en rango [0; 10]; y b) Mismo controlador de a) pero probado con ruidos en rango [0; 5].

Para los controladores de figuras 7-a) y b) el cambio de posición se produce cuando el agente comienza a percibir el objeto en más de un sensor. De esta manera, el controlador comienza a percibir el tipo de objeto y a definir su estrategia de captura o evasión que se manifiesta por un incremento de velocidad de sus motores. Por esta razón, el controlador *no reactivo* (con memoria de sus eventos pasados) no logra volver a capturar a tiempo las características del rombo y lo evade en la mayoría de los casos. Esto no pasa con el controlador *reactivo* que tiene éxito en más casos de captura.

El comportamiento de los agentes de las figuras 7-c) y 7-d) es un tanto diferente. En este caso, si bien el agente *no reactivo* no logra capturar efectivamente a los objetos (rombos) debido a su movimiento errático, aún conserva memoria del tipo de objeto y no lo evade. Es justamente en este caso donde el comportamiento del controlador *reactivo* es mayor puesto que al no tener memoria de los eventos pasados, puede reaccionar más rápido a los cambios aleatorios de posición del objeto y capturarlo más efectivamente.

6. Conclusiones

Se analizó la influencia de ruido externo en el comportamiento de un agente evolucionado genéticamente y corriendo sobre una arquitectura neuronal en hardware. Se tomó como caso de prueba un modelo usado extensamente en la literatura, discriminación de objetos.

Se presentó una serie de experimentos destinados a evaluar el desempeño de los distintos controladores en entornos expuestos a diferentes perturbaciones. Los controladores *reactivos* mostraron mejoras en su desempeño en escenarios sometidos a ruido neuronal y perturbaciones del ambiente frente a los *no reactivos*. Esto es, de alguna manera, esperable ya que los primeros no almacenan las influencias de los ruidos en su sistema nervioso y adaptan (o corrigen) mas rápidamente errores generados en estados anteriores. Por el contrario, los controladores *no reactivos* intentan modelar el comportamiento aleatorio del ruido conjuntamente con el aprendizaje del problema de discriminación en desmedro de su desempeño final.

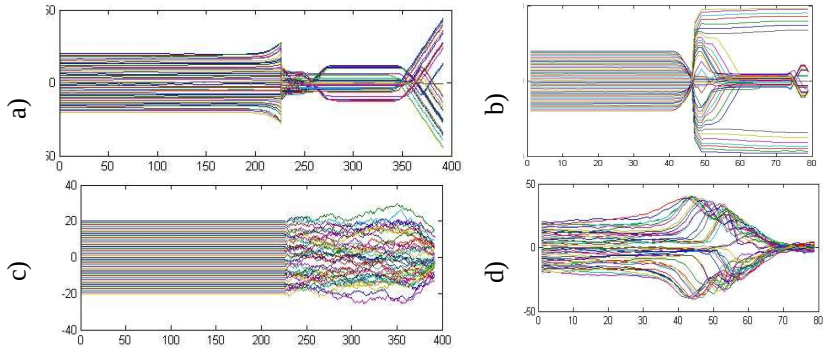


Fig. 7. Comportamiento de discriminación de controladores en escenarios con ruido ambiente. a) controlador *no reactivo* entrenado con h = 0.2 y objetos alternando su posición horizontal a la mitad de su recorrido. b) controlador reactivo y objetos con similar comportamiento al caso anterior. c) y d) Los controladores (*no reactivo* y *reactivo* respectivamente) se entrenan y evalúan con objetos con un comportamiento de zigzag a partir de la mitad de su recorrido.

7. Referencias

1. Nolfi, S.: Adaptation as a more powerful than decomposition and integration: Experimental evidences from evolutionary robotics. In P.K. Simpson (Ed.), Proc of the IEEE International Conference on Fuzzy Systems, NY: IEEE Press, 141-146, (1998)
2. Nolfi, S., Floreano, D.; Learning & evolution. Autonomous Robots, 7(1), 89-113 (1999)
3. Van Beers, R. J., Baraduc, P., Wolpert, D.: Role of uncertainty in sensorimotor control. Philosophy Transaction, The Royal Society, 357, 1137-1145 (2002)
4. Ott, H,: Noise reduction techniques in electronic systems. John Wiley & Sons (1998).
5. Balcells, J. et al.: Interferencias electromagnéticas en sistemas electrónicos. Editorial Macrombo, España (1992)
6. Di Paolo, E., Harvey, I.: Decisions and noise: The scope of evolutionary synthesis and dynamical analysis. International society for Adaptive Behavior. 11 (4), 284-288, (2003)
7. Gallardo, D., Colomina, O., Flórez, F., Rizo, R.: Generación de trayectorias robustas mediante computación evolutiva. Seminario sobre computación evolutiva: teoría y aplicaciones - SCETA 97. (1997)
8. Beer, R.: The dynamics of active categorical perception in a evolved model agent. Intenational Society for Adaptive Behavior. 11(4), 209-243 (2003)
9. Araujo, L., Cervigon, C: Algoritmos Evolutivos – Un enfoque Práctico. RA-MA Editorial, España. (2009)
10. Tosini, M.: Diseño de un procesador neuronal orientado a redes multi-etapa entrenado con backpropagation. XV Workshop Iberchip - IWS'09. (2009)
11. Fleury, P., Bofill-i-Petit, M.: Neural Hardware: beyond ones and zeros. In European Symposium on Artificial Neural Networks - ESANN'2004. (2004).
12. Xtreme DSP Design Considerations, User Guide. UG070 (V1.2). Xilinx, `www.xilinx.com/support/documentation/user_guides/ug070.pdf`
13. Xtreme DSP Design Considerations, User Guide. UG073 (V1.2). Xilinx, `www.xilinx.com/support/documentation/user_guides/ug073.pdf`