An Associative Memory for Autonomous Agents

Alexander Bisler
Fraunhofer Institute for Computer Graphics
Darmstadt, Germany
Email: abisler@igd.fhg.de

Abstract— **To improve the abilities of a behavior based autonomous agent a biologically inspired memory is introduced. The memory content is built up from scratch. The data containers, called mnemograms, are general and flexible enough to handle different kinds of data. Using a reinforcement mechanism valuable mnemograms are separated from those of lesser value. Mnemograms are stored in three hierarchically ordered layers to separate new from older ones. This is important for organizing search runs. The data containers are linked together, thus, a first level of semantical information is created. A first simple behavior takes advantage of these structures. It enables the agent to learn to distinguish between food particles and other particles.**

I. Introduction

When developing autonomous agents, one of the key issues is the interaction with their environment ([1]). Important parts of the agent's design are sensors to "see" the environment, and actuators to act in it. Using sensors and actuators and a simple "nervous system", the agent already can show a specific behavior. To improve the agent's behavior, its abilities must be extended by adding the capability to interpret the sensor data and finally to learn. Here, learning is meant as the process of deriving new knowledge from any given information. Learning implies remembering data as a comparison to other new data.

Problems arise when the memory capacity reaches its limits. What should be forgotten/deleted to make room for new data ? What should be preserved, which data are considered valuable ? Tradeoffs have to been made between continued learning and storing old data. Grossberg described this situation as the *stability-plasticity dilemma*. He developed the Adaptive Resonance Theory (ART) as a theory of human cognitive information processing ([2]).

To tackle this dilemma some solutions on different levels of the information processing pipeline are suggested in this paper. The first step is to ignore noisy input from the sensors and only handle events which exceed an "attention-threshold". Then a reinforcement mechanism and a hierarchical segmentation of the memory is introduced to separate data which are often used or recognized by the system from those which are seldomly touched. This system can be refined in the future by using informations from (e.g. behavioral) algorithms which interpret the data and give them a meaning.

A memory structure is presented to handle sensor and internal data. This structure is general and flexible enough to support future algorithms to work with these data and to let them extend the agent's memory.

First, the simulation setup and the basic design of the agent is explained. Then a closer look on the agent's sensors is taken, since they are the first part in the information processing pipeline. Following, the structure of the memory is described, the hierarchical layers, the data containers, and the associations. Then the construction of mnemograms and associations is explained, and finally their usage.

II. The Simulation Setup

A simulation was implemented which provides a 3D underwater environment in which the agent swims around. The water is filled with two kinds of particles: plankton (food) and inedible particles. The agent's behavior is kept simple in this scenario to focus on the functionality of the memory. Basically, the agent swims around and each time it bumps into another particle, it tries to eat it. Later, the agent tries to distinguish which particles are edible and which are not. The simulation is frame-based.

A. The Agent's Design

The agent is constructed of sensors, actuators, and a simple nervous system (here called: central control unit) which contains the memory and controls the behavior. Its body is a sphere[1].

Following the sense-compute-act paradigm the central control unit handles the incoming data from the sensors and passes them to the memory. Then the modules, concerning the behavior, are executed.

The actuators (e.g. fins) are not implemented in detail. The agent just knows how to swim.

B. The Sensors

The agent forms an autonomous system. The only way for data to enter that system is through the virtual sensors ([3]). They are the only source of information about the world outside.

The agent has a tactile sensor (its skin) and an olfactory sensor (both rely on collision detection with other particles - olfactory particles are very small ones).

These sensors can be described as *passive sensors* in comparison to *active sensors* which can be controlled by the agent - e.g. an eye, which can look around, or a hand which can reach out to touch something specific.

On a collision the agent's tactile sensor receives not only the boolean information that a collision occurred but also at which point on its surface/skin. Additionally, a so-called "signature" is transmitted ("perceived") from the particle,

[1] to allow a fast collision detection

which was touched, to give it some more complex data for learning (Fig. 1 (a) and (b)). This signature is an array of scalars (in this scenario 30 floating point numbers) which are manually chosen by the programmer, but slightly modified by a noise generator. Collisions are registered all over the agent's surface.

The olfactory sensor is based on the same principles: the (small) olfactory particles are detected using the same collision detection algorithm. But the sensitive areas are limited in size and there are four different areas to recognize from what direction a sent comes (left, right, above, or below). The olfactory signatures of the "smelled" particles are different to the tactile signatures, but can be handled the same way.

Other kinds of sensors might be added later.

C. Data-Preprocessing: Self-Adjustment & Filtering

The kind of data the sensor transmits to the central control unit underlies technical constraints (or implementation constraints). Nevertheless, the handling of the data has to be general and flexible since a changing environment affects the quality of the information.

In the beginning, when the agent wakes up (or comes to life), the sensors first have to adjust themselves to the given environment.[2] This is simply done by measuring the incoming data for some time and calculating the average. This average-signature is then stored and all other received signatures are compared with it (Fig. 1 (c) and (d)). Only if the difference exceeds a (technically derived) threshold, the central control unit is informed that something changed, something happened[3]. By this simple filtering, it can be avoided that the sensor transfers whatever data it measures.

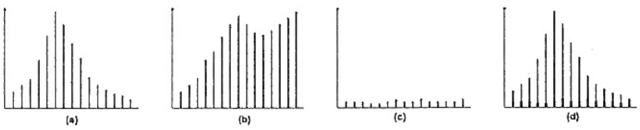

Fig. 1. Signatures: (a) of a plankton particle, (b) of an inedible particle, (c) normal input when nothing touches the skin, (d) difference between the average input and the tactile input of a plankton particle; denoted by the thinner lines above the thick line.

After the self-adjustment phase, the agent continues to modify the stored average-signature to keep track of slight but steady changes in the environment[4].

III. The Memory

The memory's atomic building blocks are data-containers which are called *mnemograms*. These can be linked together by *associations*. In the beginning, the memory is empty.

[2]When you wake up while the sun shines, your eyes adapt to the bright sun light. When you come into a dark room, your eyes have to adapt to the darkness.

[3]In the case of a non-event during a frame, an "empty" signature, containing some fuzzy data, but without any high peaks, is created to be fed into the sensor - simulating the feeling of the water around the agent.

[4]Even if the daylight becomes darker over hours, the eyes obviously steadily adapt to the changes in brightness.

A. Hierarchical Organisation

The biologically inspired memory is divided into three different layers:

- the ultra short-term memory (USTM),
- the short-term memory (STM), and
- the long-term memory (LTM).

The USTM is supposed to hold only a few mnemograms. The STM has a larger capacity and the LTM can hold even more mnemograms. New mnemograms are stored first in the USTM. Older and more often used mnemograms are transferred to the STM and later to the LTM. How they are transferred from one level to the next is described in detail in subsection *C*.

The division into three layers helps to organize the search space for algorithms which have to find specific mnemograms. By searching first through the USTM, it might be possible to find a valid mnemogram in a very short time. Checking all stored mnemograms repeatedly would be too slow. Considering a memory with a vast capacity, this would paralyze the agent, prohibiting it to quickly react to changes in its environment.

B. Mnemograms

A mnemogram is basically a container which can hold different types of data. It provides for each data package some common information as for example a timestamp or a location (if given).

Timestamps and locations are of course not absolute but relative. The agent does not have access to the CPU's clock, but keeps track of its own life time. Location coordinates are never global. The agent has no access to the scenegraph. These coordinates describe either points on the agent's surface/skin - i.e. they are part of the local coordinate system - or have to be interpreted as a relative position - e.g. the distance to a perceived object.

The types of data which have to be handled by an agent are quite limited, since the agent forms a closed system:

- each *sensor* provides a specific type of *data*
- *internal conditions* - such as hunger, pain, etc.
- *actions* performed by the agent
- *abstract data*, e.g. timestamps to provide temporal information

These data can be organized in three different types of mnemograms:

- *master-mnemograms*
- *delta-mnemograms*, describing changes or differences
- *class-mnemograms*, describing a group of mnemograms representing a pattern

In case of sensor data, not only the original ("measured") data is stored in a mnemogram but also delta-values which describe the difference to the "normal" sensor state. They are as important as the original data - maybe even more important. Consider the data from a visual sensor. "Seen" objects appear quite different during daytime and evening, when it is darker. But the difference of the lighter and the darker instance to their background - describing the shape of the object - are more comparable and can help the agent to recognize similar objects in different environmental situations ([4]).

Delta-mnemograms can also describe the change of an internal condition.

C. Validity and Importance over Time

Over time, the memory is flooded with mnemograms and at some point a decision has to be made, which mnemograms to delete to make room for newly incoming mnemograms. This is done using a *reinforcement mechanism.*

Each mnemogram has a *weight.* The initial value is larger than 0. Over time this value decreases, finally reaching 0. But each time a mnemogram is used by an algorithm (a behavior) its value is increased to mark it more important than those mnemograms which are never touched again.

When mnemograms have to be deleted, those with the smallest weights are chosen, for they were seldomly or never touched and can be considered least important to the agent.

These weights can also be used in other ways. Mnemograms which exceed a specific limit are transferred from the USTM to the STM. If the weight exceeds another threshold, the mnemogram is transferred from the STM to the LTM.

Since the mnemograms in the STM and the LTM have already proven to be more valuable, the decreasing rate of their weights is lowered. This allows them to stay longer in memory before reaching a value of 0.

D. Associations

An association links two mnemograms together; setting them in a specific relationship providing thus semantical information. The relation of both mnemograms can be seen as symmetrical (symm.), asymmetrical (asymm.), or hierarchical (hier.).

There are different types of associations possible within the agent's memory:

- spatial
  - a mnemogram has a very close relative position to another one (symm.)
  - a mnemogram has a different position than another one (asymm.)
  - a mnemogram is a part of another one (hier.)
- temporal
  - two mnemograms have the same timestamp (symm.)
  - a mnemogram follows another one (asymm.)
- content-related
  - a mnemogram containing sensor data of similar content to the sensor data of another mnemogram[5] (symm.)
- action-related
  - a mnemogram contains the same action type as another one (symm.)
  - a mnemogram containing an action is linked to a mnemogram describing the reaction (e.g. the change of an internal condition) (asymm.)
  - a mnemogram containing an action is linked to a mnemogram containing data considered as an object of the action (hier.)
- identical object
  - two mnemograms are linked together which can be considered to belong to the same object (symm.)
  - a mnemogram represents an appearance of an object described by another mnemogram (asymm.)
  - two mnemograms are linked together which can be considered to belong to the same object, but one is a part of the other one (hier.)
- instantiations of a class of patterns/objects
  - a mnemogram belongs to a class of data-patterns (hier.)

[5]only data from the same type of sensor can be compared

These association types provide a first set, which might be expanded later, when semantical relationships of a higher level can be provided and used.

To distinguish more important associations from less interesting ones, each association has a *weight* - just like the mnemograms - and the same *reinforcement mechanism* is used to increase the values which otherwise decrease over time.

An association is not deleted by the garbage collection. It only is deleted if one of the two mnemograms it connects is removed from the memory.

The weight of an association is increased each time it is used by an algorithm in a useful way.

IV. Construction of Mnemograms and Associations

Each frame of the simulation, sensors receive data from the environment. After filtering, some data packages reach the agent's central control unit. They consist of master- and delta-mnemograms which are connected with an 'identical-object' association (Fig. 2).

Each behavior(-layer), that triggers an action, constructs also mnemograms to store these actions. If internal conditions are changed as a result of these actions, their change is stored in a delta-mnemogram linked to the action's mnemogram.

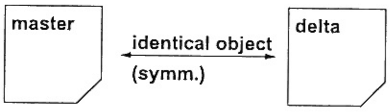

Fig. 2. A master- and a delta-mnemogram containing sensor data. Since both mnemograms contain data belonging to the same object, they are linked accordingly.

A. Connecting New Mnemograms

Each simulation frame, the memory checks whether new mnemograms were stored. These mnemograms are linked with temporal associations to show that they were created at the same time (Fig. 3).

To create a connection between mnemograms that follow each other in time an abstract mnemogram, containing only a timestamp, is generated. This one is first linked with all new

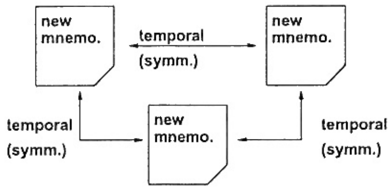

Fig. 3. Mnemograms arriving in the memory at the same time are connected with temporal associations.

mnemograms and then with the previous time-mnemogram. This way, the many connections between each mnemogram of one time frame to each mnemogram of the previous time frame can be avoided (Fig. 4).

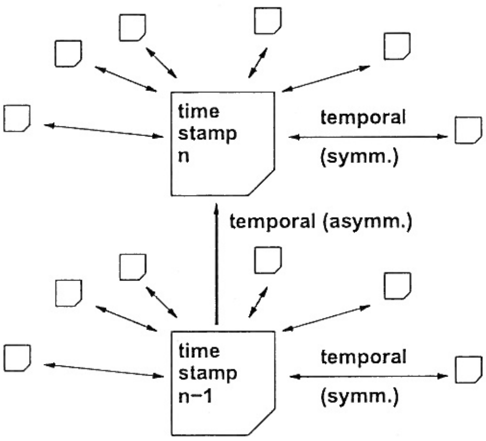

Fig. 4. Mnemograms of different time frames are connected via abstract mnemograms, thus forming a temporal chain.

B. Comparing Mnemograms

Further, the new mnemograms are compared with every other mnemogram stored in the USTM at this time[6] to derive more information. Spatial[7] (S), temporal (T), and content-related[7] (C) attributes are compared. The results are interpreted by checking whether the similarity falls within (technically derived) limits which suggest that the compared attributes are identical.

Fig. 5 shows the interpretation for mnemograms constructed for tactile input. The temporal attribute is the timestamp of the mnemogram, the spatial attribute denotes the point on the surface of the agent, where the collision took place, and the content is the tactile signature.

Depending on the combination of matching attributes (a 'T' for example denotes that the temporal attribute is identical or very close, a '!T' denotes that it's not) the found (older) mnemogram is connected with the new mnemogram.

If all three attributes are identical or similar enough, the mnemograms belong to the same object. But even if only the temporal and the spatial attributes are identical they can still be considered to belong together. Based on this assumption, the limits used to identify identical content-related attributes for this object can be extended.

If the collisions took place at different locations but at the same time and with similar tactile signatures, the mnemograms could belong to different objects and only are linked with a temporal and a content-related association.

If similar signatures are registered at the same location but not at the same time, they are only linked with content-related association, since the spatial information of the tactile sensor is not so significant.

If only one of the attributes is valid at a time, the mnemograms are only linked with one of the basic association types.

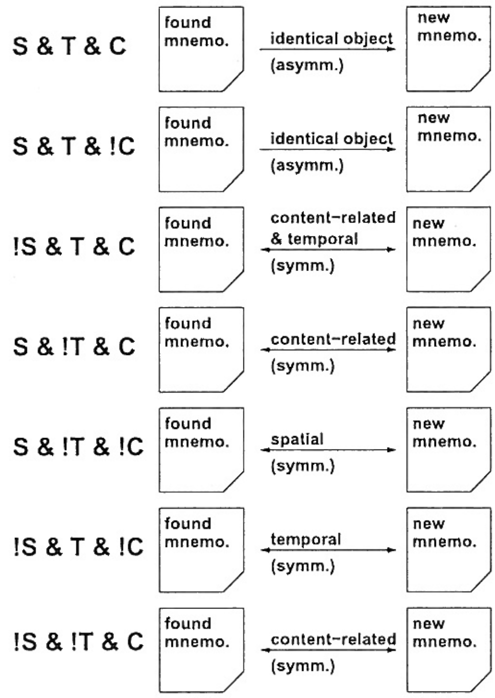

Fig. 5. Construction of associations for tactile mnemograms with similar attributes. S, T, or C denotes that the spatial, temporal, or content-related attribute falls with limits that are interpreted as identity. !S, !T, or !C indicates that the attribute is not similar enough.

V. Using the Memory

A. Searching for Temporal Patterns

As a first application a pattern searching algorithm has been implemented. The algorithm searches for chains of mnemograms which can be found more than one time. There's only a temporal constraint ensuring that the mnemograms lie close together in time. Each time new mnemograms enter the USTM the algorithm tries to find a chain starting with one of these new mnemograms which can be found again in the memory.

The algorithm is not meant to find the global optimum, but to find a local optimum within a short period of time. So the

[6]In case the USTM is empty, the search might be expanded to the STM. But the search space should be limited to avoid long processing times.

[7]if possible

search space has to be narrowed. This is done by limiting the number of mnemograms which are checked.

The first examined mnemogram is compared to a group of older ones. If a similar mnemogram can be found the algorithm tries to recursively extend both chains by finding another pair of similar mnemograms.

Given the first mnemogram a group of potiential successors is retrieved by following the associations. This group is sorted by its creation timestamps - youngest first. Mnemograms with the same timestamp are sorted by their weight. The successors for the mnemogram of the other chain are retrieved likewise. Examined mnemograms have to be marked as touched to avoid endless loops.

To narrow the search space again, the temporal distance between two successing mnemograms is limited, ensuring that only events are linked together which occurred within a short period of time.

Each time a potential successor is examined the search space is extended by retrieving its linked mnemograms. While inserting the new mnemograms into the list the temporal sorting is continued. Since all mnemograms are at least linked by temporal associations the whole memory could be searched.

As an example scenario, particles (called P), which release olfactory particles, were placed in the agent's environment. While the agent wandered around it smelled this specific scent (called S) - i.e. its olfactory sensor detected a collision with the olfactory particles. The sensor then gets a specific signature which is stored in memory[8]. The closer the agent approached a particle P the more often an olfactory particle was detected. From time to time, the agent also bumped into a particle P itself.

The pattern finding algorithm searched then for event-chains that appeared more often. Mostly it found chains consisting of collisions with olfactory particles followed by collisions with olfactory particles and so on. But it also found that a particle P can be touched just after smelling the scent S (Fig. 6). This information enables the agent to make predictions about the future and to broaden its temporal horizon ([5]). If the particles P would be food, the detection of the scent S could be used to hypothesize about something edible in the near vicinity.

Since the search algorithm has to compare a lot of mnemograms to find new patterns, its complexity his too high making it not useful for the time the agent is awake or too busy. But it is still sufficient to use a "sleeping phase" for these rountines or phases when nothing happens and processing time can be spared. When new mnemograms arrive in the USTM, it is still possible to match them with the patterns already found within a short time.

B. Refining Data Structures

Each time a chain of mnemograms, starting with a newly arrived one, matches with a pattern, the weights of these

[8]After the collision the olfactory particle is removed.

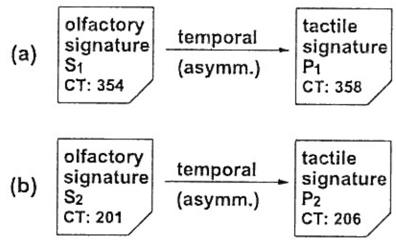

Fig. 6. Reoccurring patterns with a temporal constraint. The search starts with the youngest mnemogram P_1 finding a similar but older mnemogram P_2 (CT := creation time). Following the associations into the near past another pair of similar mnemograms S_1 and S_2 can be found. The chains (a) and (b) are then interpreted as representations of the same class of patterns.

pattern-mnemograms are increased. This way even the different classes of patterns are separated into those, which are often recognized, and those, which are less significant.

This first level of semantical structures is created with only statistical means. Thus, they only show everyday knowledge as in the scenario described above: after detecting an olfactory particle it often happens that another one is detected and so on. The next evaluation of the significance of patterns has to be done by the behavioral algorithms which can give a meaning to these data. For example a routine concerned with finding food would reinforce patterns which contain mnemograms representing something edible.

C. Structured Search Space

The pattern finding algorithm described above sorted the examined mnemograms by their timestamps because it tried to find temporal chains. Other algorithms without this temporal constraint can make use of the association weights.

Since they have different weights, they can be used to sort the list of unchecked mnemograms. Each time a mnemogram is examined, the connected mnemograms are immediately inserted into the list, which is then sorted again. This way, the search run is not breadth-first anymore. But the way the search is guided through the interconnected mnemograms is more semantically based.

The division of the memory into three layers helps also to organize the search space. A search run usually starts with the youngest mnemograms, those in the USTM and the STM, hoping that the needed information is not much older.

D. The Agent in Action

Another scenario was implemented to use the memory. The agent's task is to swim around to find food. Two kinds of particles drift around: plankton and inedible particles. When colliding with them, the agent perceives for each kind a specific tactile signature, which varies only slightly for particles of the same class, but is more distinguishable between the two classes.

In the beginning, the agent has no knowledge of what the signatures look like and which class of particles provides which signature-type. So, it has to swim around and eat whatever it can find.

Each time the agent gorges a particle, this event is stored as a gorge action. For the gorge action and the change of the hunger condition mnemograms are constructed. The tactile input is automatically stored. The weight of these mnemograms is significantly increased, since they belong to something that is important to the agent. The mnemograms are then connected with the types of associations as shown in Fig. 7.

After gathering events in which some of the eaten particles satisfied the hunger and some did not, the agent can calculate the average for each type of signature.

This is done by searching for mnemograms which contain a gorge action. Following the asymmetrical action-related association, the delta-mnemogram describing the change of the hunger condition can be found. Following the hierarchical action-related association, the tactile input is retrieved. If the hunger condition improved, the tactile input is interpreted as belonging to an edible particle, otherwise not.

After calculating the average signature for each case, a newly incoming tactile signature is compared to both of them. The agent decides whether to eat the touched particle or not depending on the similarity to one of the average signatures.

The agent learned after a few experiences, what to eat and what not.

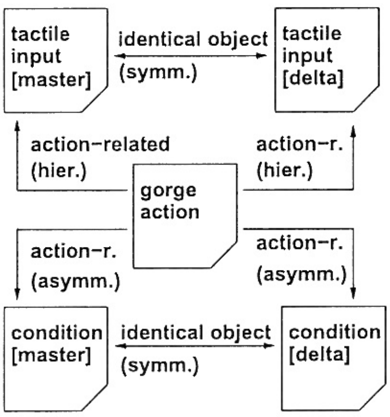

Fig. 7. Data structures built up for each gorge action.

VI. Discussion

Based on gathering data and refining them by using reinforcement mechanisms a biologically inspired memory structure could be established which is general and flexible in design. The introduced data structures provide a first level of semantical information. Algorithms working with them enabled the agent to learn from past experiences.

While connecting mnemograms it has to be ensured that all mnemograms are connected somehow. No separated mnemogram-clusters should be created. Otherwise it is not possible to reach them by following associations of other mnemograms. Since all new mnemograms are connected by temporal associations, this problem does not arise in the beginning. The time-mnemograms, which form the time line, should not be deleted unless they have no more links to other mnemograms of this time frame. In case one of them is deleted a new association has to be constructed to linked its predecessor and successor - ensuring that the time line is not interrupted.

In some simple cases mnemograms were connected to show that they belong to the same object. But the general ability to recognize a unique entity requires better sensors. They need an extended range[9] to detect distant objects. And the agent should control their focus, to be able to follow objects. Only if detailed information are provided, a problem like "identity" of an object can be approached.

In the "smelling"-scenario the agent was sometimes hit by too many olfactory particles. This flooded the memory with the same kind of events without providing any useful new information. Instead of handling these closely related events as single peaks they should be combined and represented by only one or two mnemograms which are valid of a longer time interval.

VII. Future Work

Sensors of various types (e.g. active vision [6]) are needed. Then, by examining data coming in from different sensors at the same time more useful information about objects can be gained.

The simulation has to be extended to provide more stimuli for the agent and to provide scenarios rich of information which can be perceived by the agent's sensors.

After establishing mnemograms for basic data-blocks, meta-types handling groups and classes of patterns need to be developed. And after finding patterns in the mnemogram network, the advantages of such a categorization need to be fully exploited.

References

[1] S. W. Wilson, "The animat path to AI," in *Proceedings of the first international conference on simulation of adaptive behavior on From animals to animats*. MIT Press, 1990, pp. 15–21.

[2] S. Grossberg, "Adaptive pattern classification and universal recoding: I. Parallel development and coding of neural feature detectors," *Biological Cybernetics*, vol. 23, pp. 121–134, 1976, reprinted in Anderson and Rosenfeld, 1988.

[3] N. Magnenat, N. M. Thalman, and D. Thalmann, *Artificial Life and Virtual Reality*. John Wiley & Sons, Inc., 1994.

[4] S. Marsland, U. Nehmzow, and J. Shapiro, "Environment-specific novelty detection," in *Proceedings of the seventh international conference on simulation of adaptive behavior on From animals to animats*. MIT Press, 2002, pp. 36–45.

[5] C. L. Nehaniv, D. Polani, K. Dautenhahn, R. te Beokhorst, and L. Canamero, "Meaningful information, sensor evolution, and the temporal horizon of embodied organisms," in *Proceedings of the eighth international conference on Artificial life*. MIT Press, 2003, pp. 345–349.

[6] D. Terzopoulos and T. Rabie, "Animat vision: Active vision in artificial animals," in *Int. Conf. on Computer Vision*, 1995, pp. 801–808. [Online]. Available: citeseer.nj.nec.com/rabie99animat.html

[9]like a visual sensor