Anticipatory Robot Navigation by Simultaneously Localizing and Building a Cognitive Map

Yoichiro Endo
endo@cc.gatech.edu

Ronald C. Arkin
arkin@cc.gatech.edu

Mobile Robot Laboratory, College of Computing,
Georgia Institute of Technology, Atlanta, GA, 30332-0280

Abstract

This paper presents a method for a mobile robot to construct and localize relative to a "cognitive map", where the cognitive map is assumed to be a representational structure that encodes both spatial and behavioral information. The localization is performed by applying a generic Bayes filter. The cognitive map was implemented within a behavior-based robotic system, providing a new behavior that allows the robot to anticipate future events using the cognitive map. One of the prominent advantages of this approach is elimination of the pose sensor usage (e.g., shaft encoder, compass, GPS, etc.), which is known for its limitations and proneness to various errors. A preliminary experiment was conducted in simulation and its promising results are discussed.

1. Introduction

Suppose that an office robot is about to be sent on errands to deliver some document from Office A to Office B. Office A and Office B are on different floors; hence, the robot has to take an elevator. A practical question in order to implement such a robotic task would be, "How should Office A, Office B, and the elevator, as well as the actions (e.g., taking the elevator) be represented within the robot?" One may suggest using a map. However, many conventional maps contain geometrical information only, and do not directly address the question of which actions should be organized at any given time.

Of course, the actions required for navigation may be dealt separately from the map, treating them as a classic AI problem. In this paper, however, a means for incorporating both forms of knowledge (behavioral and spatial) into a single representational structure is investigated. In the remainder of this paper, this representational structure is referred to as a *cognitive map*, distinguishing it from a conventional map that only stores geometrical information.

Kuipers [6] asked the questions of how the cognitive map should be designed if it was intended for use by a mobile robot. His suggestions included:

- The cognitive map should be constructed from a number of different frames of references since fitting all the geometric information in a single (world coordinate) frame requires highly biased interpolations.
- The knowledge of how to get from one point to another should be represented adequately in the cognitive map as it is vital to navigation.

This led Kuipers and Byun to propose a mapping strategy that is based on a qualitative method [7]. By using a hill-climbing method, *distinctive places* in the environment are detected. These *distinctive places* are stored in a topological network and the linking relationships among them are described by a control strategy. Lee [8] implemented this method on a real robot.

Similarly, Mataric proposed a topologically organized distributed map [12]. Each node in the topological network represents a landmark, and nodes can communicate among themselves through spreading activation. Moreover, when a goal object is given, each node can suggest a real-time procedure for the robot to navigate through the environment in order to move to the goal. This method is inspired by how hippocampus of a rat operates [13].

A probabilistic approach can be also incorporated into a topological map framework. For example, Koenig and Simmons [5] implemented a POMDP-based navigation architecture for a mobile robot. Given a predefined topological map, the system can estimate the current state while the robot navigates in an indoor environment using this probabilistic method, and it can suggest the most rewarding action to take in order to reach a designated goal. Koenig and Simmons' system assumes that the topological map is given (since the emphasis is on the policy mapping). On the other hand, research by Thrun et al. [22] generates a map as it moves through the environment. It utilizes both topological and metric information. Although manually picked (instead of using the hill-climbing method), *distinctive places* were utilized to solve the correspondence problem of localization, and the Expectation Maximization (EM) method was used to generate a map of a large scale environment that contains a loop structure.

The representation of the cognitive map in this paper alternatively utilizes *episodic* memories, where the concept was inspired by the recent biological findings [3][26] (explained briefly in Section 4). The goal of this paper, however, is not to implement or validate the fidelity of the underlying biological model, but rather it is an examination of how the past experiences of a *robot* can be stored in a meaningful way, so that the it can "localize" itself relative to its past experiences, anticipate a future event, and then select and coordinate behavioral actions appropriately to react to these anticipatory events.

2. Cognitive Map for Anticipatory Behavior

A model of a cognitive map that allows a robot to anticipate future events based on its past experience is described in this section.

2.1. Cognitive Map That Encodes Episodes

We chose *MissionLab* [11], a software tool that implements a behavior-based robotics system [1], as the target platform to implement the cognitive map model. Because of the tight coupling of perception and action that a behavior-based robotics system provides, constant streams of both sensor readings, denoted here as z, and behavioral motor commands, denoted as u, are considered to be always available.

Suppose then that there is a way to detect novelty in the environment (discussed in Section 2.2). The instance when the robot detects this environmental novelty is considered as an "event", denoted as e_i (where the subscript $_i$ is some particular instance i), and e_i contains spatial and behavioral information as well as a tracking number n_i (the number of the events stored thus far in memory). In other words, e_i is a *set* of the sensor readings z_i, the motor command u_i, and n_i at some instance i:

$$e_i = \{z_i, u_i, n_i\} \tag{1}$$

Notice that since the detection of novelty depends on the nature of the environment, e_i is independent of time. However, suppose that a number of events have been detected, then the order of the event sequence, $e^i = (e_1, e_2, \dots, e_i)$ is temporal, and not spatial (the superscript ^{i} is used here to refer to all the instances up to the instance i).

This sequence of the events constitutes an episode, denoted here as E. In other words, E is an ordered *i-tuple*,

$$E = (e_1, e_2, \dots, e_i)$$

The robot records all of the experienced episodes in its episodic memory, and this collection of episodes is hereby defined as a cognitive map. In other words, the cognitive map C can be described as an ordered *tuple*:

$$C = (E_1, E_2, \dots, E_K)$$

where K is a total number of the episodes experienced by the robot.

In summary, whenever the robot is active, it constantly builds a cognitive map by accumulating the experienced episodes in its memory, where the snapshots of the sensor readings and the motor commands whenever novelty in the environment is detected are stored in the episodic memory. "When does an episode begin?" and "when does it end?" are questions that should be fully investigated. Tentatively, however, it is defined that a new episode begins when the robot starts a new predefined mission (a behavioral sequence described with a finite state acceptor), and ends when that mission is terminated.

2.2. Event Detection

As mentioned above, a snapshot of the sensor readings and behavioral motor commands are recorded whenever novelty in the environment is detected, and here this snapshot is called an event. The robot is designed to register such event based on our assumption that when the environment is novel, there is a discontinuity between the current environment and the environment just before, and thus it is worth remembering the characteristics of this new environment for the future use.

The information about the environment can only be perceived by the robot's available sensors. Thus, in order to detect the novelty (unpredictable characteristics) of the environment, the robot has to constantly predict the set of incoming sensor readings; whenever the robot's prediction is incorrect, the novelty of the environment (i.e., event) is considered to be detected. The novelty feature should only depend on the characteristics presented by the environment itself. In other words, the robot should detect exactly same novelty features in the environment whether it is moving fast or slow.

The prediction of the sensor readings is implemented here using Sutton's *Temporal Differencing* or *TD*(λ) [19]. While this method is usually employed to predict the rewards, here, it is used to predict each of the sensor readings (i.e., the perceivable environment).

Suppose that the robot is equipped with two types of typical sensors: an array of sonar sensors and a CCD camera. The array of the sonar sensors consists of N transducers mounted evenly on the circumference of the robot, and thus it provides N readings (distance to a closest object) per full-scan, denoted as $S_1, S_2, \dots, S_N$. The output of the camera is connected to Newton Cognachrome board that provides detection of color blobs. The 180-degree camera view is divided into M segments in azimuth, and thus it provides M readings (distance to a closest color object) per instance, denoted as $C_1, C_2, \dots, C_M$. Let us also suppose that the robot executes a motor command u, which is a behavioral output described with a 2-D velocity vector $\langle u_d, u_\theta \rangle$; u_d is the magnitude and u_θ is the direction of the velocity (explained further in Section 2.5).

The schematic representation of the sensor prediction is shown in Figure 1. Here, all of the variables above are combined as a vector x_t (i.e., $x_t = \langle S_1, S_2, \dots, S_N, C_1, C_2, \dots, C_M, u_d, u_\theta \rangle$) after being normalized, and all of the elements in the vector contribute to predict each of the sensor readings for the next time cycle. For example, prediction of the next sonar reading for S_1, denoted as S'_{1t+1}, is done by computing a linear function of weights w_{S1t} and vector x_t:

$$S'_{1t+1} = \sum_{a=1}^{N+M+2} w_{S1t}(a)\, x_t(a) \tag{2}$$

While this computational method resembles a conventional neural network, it should be noted that the "network" used here does not have hidden layers, and the output units do not have any activation function (such as a sigmoid function, a constant threshold, etc.) as neither of them were implemented in the original *Temporal Differencing* method [19].

At each time cycle, the weights are updated using the *TD*(λ) update rule. While *TD*(λ) is typically used to predict a delayed reward, since the stream of the sensor readings are assumed to be always available *TD*(λ) is employed as a single-step prediction. For example, at time t, the increment of the weights for the sonar reading S_1 is computed by:

$$\Delta w_{s1t} = \alpha(S_{1t} - S'_{1t})\sum_{k=1}^{t}\lambda^{t-k}\nabla_w S'_{1k} \tag{3}$$

Here α is a learning rate, λ^k is an exponential weighting factor, and the gradient $\nabla_w S'_{1t}$ is a partial derivatives of S'_{1t} with respect to the weights. Because S'_{1t} is a linear function of w_{s1} and vector x_t (Equation 2), the value of the gradient is simply x_t. Thus, Equation 3 can be rewritten as:

$$\Delta w_{s1t} = \alpha(S_{1t} - S'_{1t})\sum_{k=1}^{t}\lambda^{t-k}x_k \tag{4}$$

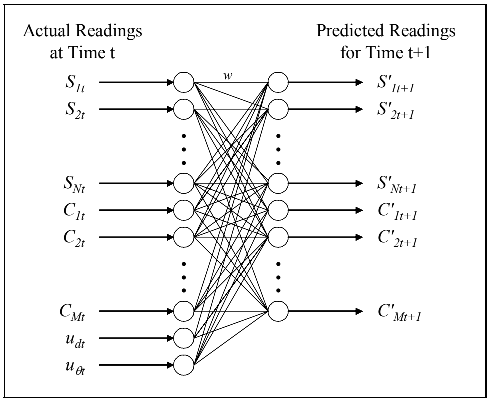

Figure 1: Prediction of sensor readings.

At each time cycle, a *root-mean-square* (RMS) of the errors between the predicted sensor readings (based on the previous time cycle) and the actual sensor readings are computed, and it is used as guidance for creating a new event e_i. For example, the graph in Figure 2 shows the RMS prediction error when a robot moved from one end of a corridor (left in the figure) to the other end (in simulation). The prediction error generates number of spikes. A peak of each spike is, here, considered as occurrence of a new event. As it can be observed from the figure, the spikes (or events) seem to capture salient features for the robot to navigate in the environment, such as doors and the corridor junction.

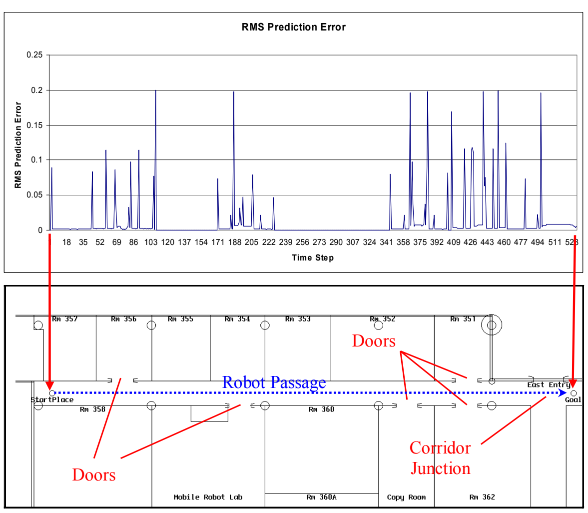

Figure 2: Sensor prediction error and the environment (simulation). The peak of each spike is considered as occurrence of a new event.

In terms of the event detection, Lewis [9] and Lewis and Simo [10] have applied a similar approach to their biped robots. In those cases, visual information (optic flow [9] and stereoscopic data [10]), along with joint angles and a gait phase, was fed into a neural network. The weights were updated by applying the Widrow-Hoff rule [25]. The robot was able to detect "novelty" in the environment while walking through it. It should be noted that the *TD*(λ) update rule is based on the Widrow-Hoff rule; it was extended in order to implement incremental learning of weights and multi-step prediction [19].

2.3. Localization

In robotics, solving the problem of building a map of an unknown environment while simultaneously identifying its location with respect to the map (SLAM problem) is considered to be one of the most challenging tasks, and it has been widely investigated in past years [21]. One of the reasons that makes this problem nontrivial comes from the nature of any physically realized robot; it has to deal with uncertainties produced by actuators, sensors, interpretation of the sensor data, accuracy of the map, initial position of the robot, and the dynamic nature of the real world [18]. As Thrun reports in his survey paper [21], all of the successful approaches to this localization and/or mapping problem today employ probabilities, which are some forms or extensions of the Bayes filer. While such probabilistic approach usually provides a means for a robot to localize itself relative to a conventional (metric) map, here, we employ the generic Bayes filter for the robot to localize itself relative to the cognitive map.

Given a sequence of sensor readings z^i and the behavioral motor commands u^i, the posterior probability of the robot being at the same event e_x in the past can be calculated by the generic Bayes filter below (Equation 8). As explained in [20] by Thrun, the equation was derived by applying the Bayes rule (Equation 5), the Markov assumptions twice (Equations 6 and 8), and the law of the total probability (Equation 7) to the posterior.

$$\begin{aligned}
& p(e_x \mid z^i, u^i) \\
&= p(e_x \mid z_i, z^{i-1}, u^i) \\
&= \eta\, p(z_i \mid e_x, z^{i-1}, u^i)\, p(e_x \mid z^{i-1}, u^i) && (5) \\
&= \eta\, p(z_i \mid e_x)\, p(e_x \mid z^{i-1}, u^i) && (6) \\
&= \eta\, p(z_i \mid e_x) \int_{e_{x-1}\in E} p(e_x \mid z^{i-1}, u^i, e_{x-1})\, p(e_{x-1} \mid z^{i-1}, u^i)\, de_{x-1} && (7) \\
&= \eta\, p(z_i \mid e_x) \int_{e_{x-1}\in E} p(e_x \mid u_i, e_{x-1})\, p(e_{x-1} \mid z^{i-1}, u^{i-1})\, de_{x-1} && (8)
\end{aligned}$$

Here, η is a scale (or normalization) factor that ensures the sum of all the possible posteriors becomes 1.

Recall that $e_x = \{z_x, u_x, n_x\}$ (Equation 1). $p(z_i \mid e_x)$ in Equation 8 is called the perceptual model and is estimated by straightforward comparison of incoming sensor reading z_i and the stored sensor reading z_x. The difference between each element of corresponding sensor readings in the vectors is *root-mean-squared*, and its negative value is fed into the exponent function. In other words,

$$p(z_i \mid e_x) = \eta_p \exp(-RMS(z_i - z_x))$$

η_p is the normalization factor of the perceptual model. The perceptual model suggests how close the current environment is to the one in the immediate past.

On the other hand, the motion model $p(e_x \mid u_i, e_{x-1})$ in Equation 8 is estimated by the following rule:

$$p(e_x \mid u_i, e_{x-1}) = \begin{cases} \eta_m \max\left(\lambda^{d-1} \exp(-RMS(u_i - u_x)), \dfrac{1}{\Gamma}\right) & \text{if } d > 0 \\ \eta_m \dfrac{1}{\Gamma} & \text{otherwise} \end{cases}$$

Here, η_m is a normalization factor of the motion model, and λ^d is an exponential weighting factor where λ is some constant and d is distance between e_x and e_{x-1} in the episodic memory (i.e., $d = n_x - n_{x-1}$). Γ is the total number of the events stored in the episode. If the motor commands perfectly match and e_x is stored as the next event of e_{x-1} in the episode, the probability will be at its maximum value. On the other hand, if the motor commands are far different, the distribution of the probability becomes uniform.

It is assumed here that the criterion for the robot being able to localize to the past event depends on the distribution of the posterior probability $p(e_x \mid z^i, u^i)$. For example, if the posterior probability is distributed around the average value (as shown in Figure 3), no localization can be made. On the other hand, if the distribution contains distinct peaks, localization can be attained (Figure 4). The criterion for determining whether the distribution contains these distinct peaks is evaluated by a threshold Θ which is calculated by:

$$\Theta = \frac{1+\kappa}{\Gamma} \tag{9}$$

where κ is a constant value, and Γ is a number of events in the episode. If the highest peak in the posterior distribution is above Θ, then it is considered that the localization is made. In other words, the robot would localize itself relative to the stored event $e_{localized}$ by solving the following equation:

$$e_{localized} = \begin{cases} \underset{e_x \in E}{\operatorname{argmax}}\, p(e_x \mid z^i, u^i) & \text{if } \underset{e_x \in E}{\max}\, p(e_x \mid z^i, u^i) > \Theta \\ \text{N/A} & \text{otherwise} \end{cases}$$

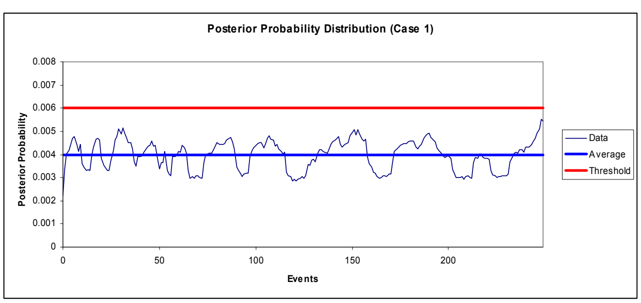

Figure 3: A case when localization cannot be achieved. As the posterior probability distributes around the average value, and none of the peaks exceeds the threshold value, the robot cannot localize relative to the past events.

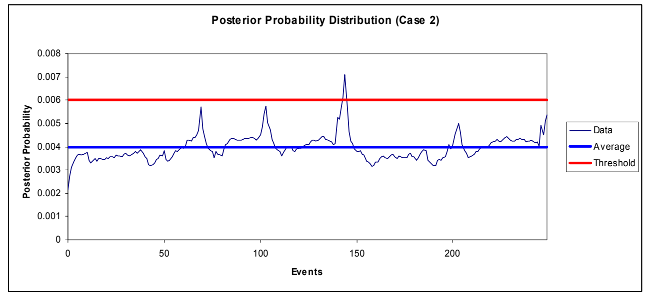

Figure 4: A case when localization is attained. As one of the peaks exceeds the threshold value, the robot can localize relative to the event (e_{146}) that corresponds to the highest peak of the posterior probability distribution.

2.4. Anticipatory Robot Behavior

In order to incorporate the episodic memory within the behavior-based architecture, a new behavioral assemblage, called *Search X*, was created. As shown in Figure 5, this behavior bears some resemblance to Brooks' subsumption architecture [2]. The top behavior is *Traject-Route-To X*, which takes the a cognitive map (set of episodes) as its input. If the goal object X is found in a stored event, say e_{goal}, and if there exists a "path" between e_{goal} and the currently localized event $e_{localized}$, the *Traject-Route-To X* will output a vector $u_{localized+1}$ that is the motor command of the event stored immediately after $e_{localized}$. Here, the criteria of a "path" existing are (1) the goal event e_{goal} and the current localized event e_x are in the same episode E (i.e., $\langle e_{goal}, e_{localized}\rangle \in E$), and (2) the target event chronologically comes after the current localized event (i.e., $n_{goal} > n_{localized}$). The question of how to connect different episodes with the path has not yet been investigated. If the robot could not be localized, or if a path between e_{goal} and $e_{localized}$ could not be established, *Traject-Route-To X* outputs a zero vector.

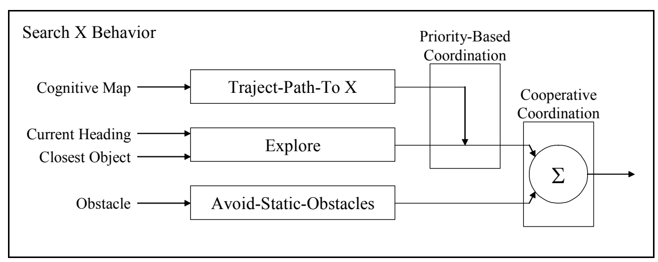

Figure 5: *Search X* behavioral assemblage.

This *Traject-Route-To X* behavior relates to Brooks' *Level 3* (*Build Maps*) and, as in his model, this behavior suppresses the output of the one below, *Explore*, through a priority-based behavior coordinator. The assemblage of *Explore* is shown in Figure 6. It was designed to explore an indoor environment by following walls by detecting the closest object. The assemblage of *Explore* consists of *Move-Ahead*, *Move-To-Object*, *Swirl-Static-Object*, and *Avoid-Static-Object* primitive schemas, which are explained in [1]. These primitive behaviors are coordinated by a cooperative coordinator vector summation mechanism. While *Traject-Route-To X* and *Explore* behaviors are coordinated with a priority-based arbiter, its output is computed by a cooperative coordinator with the *Avoid-Static-Obstacles* schema (Brooks' model, on the other hand, coordinates this level with the priority-based arbiter as well). The effectiveness of the *Traject-Route-To X* component in *Search X* behavior is sought in a preliminary experiment (Section 3).

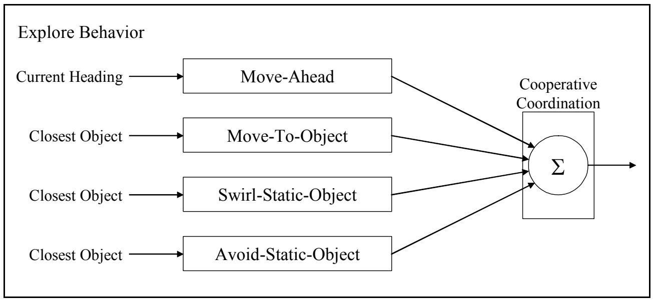

Figure 6: *Explore* behavioral assemblage

2.5. Reference Frames

At this point, it should be noted that our method never requires the use of a pose sensor (e.g., shaft encoder, compass, GPS, etc.). In fact, none of the data used in the above computation is converted into the world (or absolute) coordinate system. The only global representation being used here is the tracking number n (Equation 1). This is deliberately done so based on Kuipers' suggestion [6] that fitting all the geometric information in a single frame would require highly biased interpolations.

In our system, the sensor reading z is captured in the robot-centered (or egocentric) coordinate system, and it is used to: (1) perform localization (i.e., to compute the perceptual model in Equation 8); and (2) compute the output of the *Explore* and *Avoid-Static-Obstacles* behaviors. Neither of the tasks requires the world coordinate system. On the other hand, the motor command u is used to: (1) compute the motion model; and (2) produce the output for *Traject-Route-To X*. While neither of these tasks also requires the world coordinate system, we investigated two different approaches to represent u in the episodic memory. One obvious approach is to use the robot's egocentric coordinate system as is the case for the sensor reading z (i.e., u_θ is zero when it is pointing towards the robot's heading). Another approach is to use an environment-specific (or object-centric) coordinate system since the geometrical relationships between the robot and environmental objects are crucial to the behavior-based robotics navigation. For example, the robot may be able to treat a distinguishable landmark in the environment as a reference point, and u_θ may be measured with respect to the direction of the reference point.

For some cases, however, a distinguishable landmark may not be easily extracted from the environment (e.g., dark corridor, etc.). Alternatively, we attempted to apply the concept of principal axes in physics to identify a unique direction relative to the environment given an array of sonar readings. Consider an egocentric 3D Cartesian system. The properties of the inertia matrix with respect to its principal axis ω is described by the following equation:

$$\begin{bmatrix} (I_{xx}-I) & I_{xy} & I_{xz} \\ I_{yx} & (I_{yy}-I) & I_{yz} \\ I_{zx} & I_{zy} & (I_{zz}-I) \end{bmatrix} \begin{Bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{Bmatrix} = \{0\} \qquad (10)$$

I_{xx}, I_{yy}, and I_{zz} are called moments of inertia, I_{xy}, I_{yx}, I_{yz}, I_{zx}, I_{zy} are called products of inertia, and I is a principal moment of inertia of the system. If the system is rotated along with ω, the products of inertia vanish. Solving for ω is in fact equivalent to solving of an eigenvector problem [4]. In other words, as ω is a characteristic vector of the inertia matrix, by treating the end points of the sonar readings as "virtual particles", we can consider the direction of ω as a unique direction with respect to the formation of the environmental objects detected by the sonar sensors. The angle of principal axis ω with respect to the robot's heading (i.e., x-axis) is denoted here as φ, and its value can be obtained by the calculations below.

Suppose N virtual particles (i.e., N sonar readings) have weight that sums up to 1 as a collection, and they are distributed only on the x-y plane, the moments and products of the inertia can be computed by:

$$I_{xx} = \int (y^2 + z^2)\, dm = \sum_{i=1}^{N} y_i^{\,2}$$

$$I_{yy} = \int (x^2 + z^2)\, dm = \sum_{i=1}^{N} x_i^{\,2}$$

$$I_{zz} = \int (x^2 + y^2)\, dm = \sum_{i=1}^{N} (x_i^{\,2} + y_i^{\,2})$$

$$I_{xy} = -\int xy\, dm = -\sum_{i=1}^{N} x_i y_i$$

$$I_{xz} = -\int xz\, dm = 0$$

$$I_{yx} = I_{xy} = -\sum_{i=1}^{N} x_i y_i$$

$$I_{yz} = -\int yz\, dm = 0$$

$$I_{zx} = I_{xz} = 0$$

$$I_{zy} = I_{yz} = 0$$

For convenience, let us allow the following denotations:

$$\sum_{i=1}^{N} x_i^{\,2} = S_{xx}, \ \sum_{i=1}^{N} y_i^{\,2} = S_{yy}, \ \sum_{i=1}^{N} x_i y_i = S_{xy}$$

By substituting the values above, Equation 10 can be now rewritten as:

$$\begin{bmatrix} (S_{yy}-I) & -S_{xy} & 0 \\ -S_{xy} & (S_{xx}-I) & 0 \\ 0 & 0 & (S_{xx}+S_{yy}-I) \end{bmatrix} \begin{Bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{Bmatrix} = \{0\}$$

Note that the determinant of the left matrix is zero. Thus, we are able to compute the three possible principal moments of inertia as:

$$I_{1,2} = \frac{(S_{xx}+S_{yy}) \pm \sqrt{(S_{xx}+S_{yy})^2 - 4\,(S_{xx}S_{yy} - S_{xy}^{\,2})}}{2}, \ I_3 = S_{xx} + S_{yy}$$

The angle φ can be calculated by just using I_1 and I_2 as:

$$\varphi = \tan^{-1}\frac{\omega_y}{\omega_x} = \tan^{-1}\frac{S_{yy} - I_{1,2}}{S_{xy}}$$

Note that here φ has two possible values corresponding to I_1 and I_2. We take whichever is close to the robot's heading. The effectiveness of the two different coordinate systems (i.e., egocentric vs. object-centric) are tested in the next section.

3. Experiment

In order to verify whether the anticipatory behavior explained above could actually contribute to improve the performance of a navigational task, a simple simulation experiment was prepared. More specifically, in this experiment, the effectiveness of the *Traject-Route-To X* component in *Search X* behavior was investigated by comparing two versions of the behavior: one with *Traject-Route-To X* intact and one without it. Furthermore, as discussed in Section 2.5, the effectiveness of representing the motor command *u* in the egocentric coordinated system was compared against the one represented in the object-centric coordinate system.

The experiment was constructed to test whether the simulated robot can follow a path from a current position to a goal object based on its previous training experience. The size of the simulated robot was configured as 0.3 meters; an array of simulated sonar sensors consisted of 16 transducers mounted evenly on the circumference of the robot, and the 180-degree simulated camera view was divided into 5 segments in azimuth, providing 5 simulated Cognachrome readings. As shown in Figure 7, a simple indoor environment (T-maze), having 1.2-meter corridor width, was prepared for the experiment. The performance of the robot behavior was measured by counting the number of correct turns at the corridor junction. The robot always started around *StartPlace*, and the red object (goal object) was placed alternatively between the left and right corridors. For each trial, using predefined waypoints, two training runs were given to the robot. One training run brings the robot to make a left turn and leads it to the left corridor. The other training drives the robot to the right corridor. During the two training runs, the goal object was placed only at one side of the corridor, and, thus, the robot observed the object only once before the test. The order of the training runs was always alternated. A total of 64 tests were conducted for each condition. For each test, the run was terminated when the robot reached the end of the corridor (either the left or the right side).

In this experiment, the following constants were used for the event detection: $\alpha = 0.001$ and $\lambda = 0.1$ (Equation 4); $\kappa = 0.1$ (Equation 9). In order to simulate the real world conditions, during both training and actual testing runs, artificial noise was added to the sensor readings and the actuator output, and the initial position and heading of the robot was slightly varied. Different values for the artificial noise and the offset were chosen at each run, so that their distribution would be normal (Gaussian) throughout the experiment. More specifically, with the 95%-confidence: (1) the value of the artificial noise would be picked within 10% of the actual sensor reading or actuator output; (2) the offset of the initial position and heading would range within 0.1 meter and 10°, respectively.

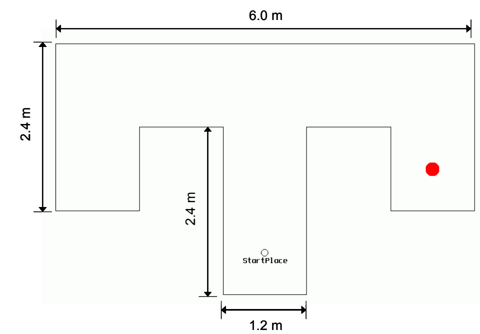

Figure 7: T-maze.

The results are shown in Figure 8. *Search X* with *Traject-Route-To X* behavior (storing egocentric motor command *u*) made considerably more correct turns (about 80% mean) when compared to the behavior without *Traject-Route-To X*, which only 50% of the time correctly choose the right turns. One-way ANOVA (computed by *STATISTICA* v6.0, StatSoft, Inc.) shows that the difference was statistically significant ($F_{1,126} = 16.869$, $p = 0.001$). On the other hand, there was no difference between the *Search X* behavior storing the egocentric motor commands and the one storing object-centric motor commands ($F_{1,126} = 0.000$, $p = 1.000$).

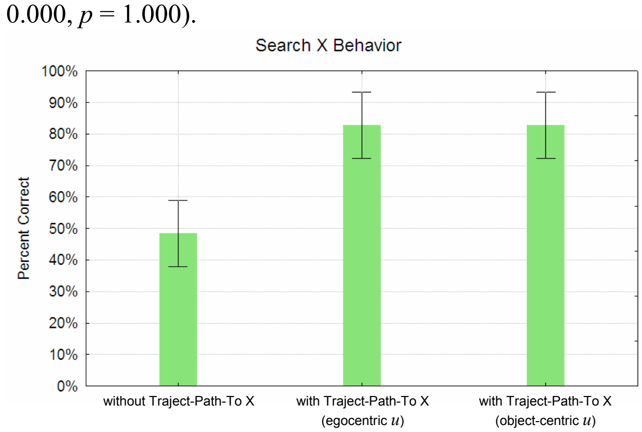

Figure 8: Mean plots of the successful turns. The narrow vertical bars around the mean values denote 95% confidence intervals.

4. Biological Basis

As mentioned earlier, the goal of this paper is not to validate existing biological models by implementing them on robots. However, biological findings did significantly influence the designed of the proposed system above. The relevant findings are briefly explained here.

The term "cognitive map" was first coined by Tolman [23] in the late 1940's to hypothesize his idea of a rat learning spatial information during a food-seeking task, which contradicted the popular psychological theory of Behaviorism at that time, where it was argued that the rat only learns through stimulus-response connectivity. A prominent study by O'Keefe and Nadel [14] suggested that the cognitive map is constructed in the hippocampus of the brain. One of the evidence cited was the notion of *place cells*; the *place cells* excite whenever the animal is in a familiar environment.

How the hippocampus recognizes the familiar environment is still debated among scientists. One school advocates that the environment is projected to a single map framework, and path-integration is employed by the animal for localizing itself in the map. In this context, high fidelity neurophysiological models of the hippocampus have been proposed by Samsonovich and McNaughton [17] and Redish and Touretzky [15]. For instance, a simulated rat implementing Redish and Touretzky's model was even able to solve the Morris water maze problem [16]. However, as their emphasis was on validating the fidelities of their models, the question of how these models would help navigating an actual robot has not been fully addressed yet. On the other hand, another school (e.g., Eichenbaum et al. [3] and Wood et al. [26]) suggests that the hippocampus stores episodes or sequences of events, each of which consists of both spatial and non-spatial information. The non-spatial information includes behaviors. As discussed in Section 1, this latter argument agrees with the points being made by Kuipers [6] for the robotic cognitive map. Therefore an episodic memory based cognitive map has been implemented here. The term "episodic memory" is first coined by Tulving [24] in order to distinguish a chronologically ordered memory from a semantic memory.

5. Conclusions and Future Work

In this paper, a method of how to construct and localize relative to a cognitive map within a behavior-based robotic framework was presented. One of the prominent advantages of this approach is elimination of the pose sensor usage (e.g., shaft encoder, compass, GPS, etc.), which is known for its limitations and proneness to various errors. The preliminary results from the simulation experiment showed that the proposed cognitive map seems to contribute to the ability of a robot to anticipate future events for navigation.

However, farther analysis of the system needs to be conducted. For example, the system must be tested on a real robot (rather than simulation). The question of computational complexity has to also be addressed, as this method currently computes full posteriors for the entire episode. It has been observed that the activity of the robot slows down drastically as the number of the accumulated events increases. Another issue that needs to be investigated is whether the current clustering of events (i.e., the event detection with $TD(\lambda)$) is adequate. This question also involves how the threshold values should be chosen meaningfully. As mentioned earlier, the questions of just when an episode starts, and when does it end, have to be resolved as well. Incidentally, the current system only allows for searching of a goal object within a single episode. The question of how to connect two different episodes should be also investigated.

References

[1] Arkin, R.C., *Behavior-based Robotics*, MIT Press, 1998.

[2] Brooks, R. "A Robust Layered Control System for a Mobile Robot." *IEEE Journal of Robotics and Automation*, 1986, Vol. 2, No. 1, pp. 14-23.

[3] Eichenbaum, H., Dudchenko, P., Wood, E., Shapiro, M., and Tanila, H. "The Hippocampus, Memory, and Place Cells: Is It Spatial Memory or a Memory Space?" *Neuron*. Cell Press, 1999, Vol. 23, pp. 209-226.

[4] Greenwod, D.T. *Principles of Dynamics*. Prentice Hall, Englewood Cliffs, NJ, 1988.

[5] Koenig, S. and R.G. Simmons, R.G. "Xavier: A Robot Navigation Architecture Based on Partially Observable Markov Decision Process Models." *Artificial Intelligence Based Mobile Robotics: Case Studies of Successful Robot Systems*, MIT Press. 1998.

[6] Kuipers, B. "The Cognitive Map: Could It Have Been Any Other Way?" *Spatial Orientation: Theory, Research, and Application*, eds. Pick, H.L. Jr. and Acredolo, L.P. Plenum Press, New York, 1983, pp. 345-359.

[7] Kuipers, B., and Byun, Y.T. "A Robot Exploration and Mapping Strategy Based on Semantic Hierarchy of Spatial Representations." *Journal of Robotics and Autonomous Systems*, Vol. 8, 1991, pp. 47-63.

[8] Lee, W.Y. *Spatial Semantic Hierarchy for a Physical Mobile Robot*. Doctoral dissertation, Department of Computer Sciences, The University of Texas at Austin, 1996.

[9] Lewis, M.A. "Detecting Surface Features During Locomotion Using Optic Flow." *Proceedings of the IEEE International Conference on Robotics and Automation*, Vol. 1, 2002, pp. 305-310.

[10] Lewis, M.A., and Simo, L.S. "Certain Principles of Biomorphic Robots." Autonomous Robots, Vol. 11, 2001, pp. 221-226.

[11] MacKenzie, D., Arkin, R.C., and Cameron, J. "Multiagent Mission Specification and Execution." *Autonomous Robots*, Vol. 4, No. 1, Jan. 1997, pp. 29-57.

[12] Mataric, M.J., *A Distributed Model for Mobile Robot Environment-Learning and Navigation*. Technical Report, MIT Artificial Intelligence Laboratory, 1990.

[13] Mataric, "Navigation with a Rat Brain: A Neurobiologically-Inspired Model for Robot Spatial Representation." *Proceedings of the First International Conference on Simulation of Adaptive Behavior*, MIT Press, 1990, pp. 169-175.

[14] O'Keefe, J. and Nadel, L. *The Hippocampus as a Cognitive Map*. Clarendon Press, Oxford, 1978.

[15] Redish, A.D., and Touretzky, D.S. "Cognitive Maps Beyond the Hippocampus." *Hippocampus*. Wiley-Liss, Inc, 1997, Vol. 7, pp. 15-35.

[16] Redish, A.D., and Touretzky, D.S. "The Role of the Hippocampus in Solving the Morris Water Maze." *Natural Computation*, Vol. 10, 1998, pp. 73-111.

[17] Samsonovich, A. and McNaughton, B.L. "Path Integration and Cognitive Mapping in a Continuous Attractor Neural Network Model." *The Journal of Neuroscience*, Vol. 17, No. 15, 1997, pp. 5900-5920.

[18] Simmons, R. and Koenig, S. "Probabilistic Robot Navigation in Partially Observable Environments." *Proceedings of the International Joint Conference on Artificial Intelligence*, 1995, pp. 1080-1087.

[19] Sutton, R.S. "Learning to Predict by the Methods of Temporal Differences." *Machine Learning*, 1988, Vol. 3, 1998, pp. 9-44.

[20] Thrun, S. "Probabilistic Algorithms in Robotics." *AI Magazine*, Vol. 21, No. 4, 2000, pp. 93-109.

[21] Thrun, S. "Robotic Mapping: A Survey." *Exploring Artificial Intelligence in the New Millenium*. Morgan Kaufmann, 2002.

[22] Thrun, S., Gutmann, J., Fox, D., Burgard, W., and B. Kuipers, B. "Integrating Topological and Metric Maps for Mobile Robot Navigation: A Statistical Approach." *Proceedings of the National Conference on Artificial Intelligence*, 1998.

[23] Tolman, E.C. "Cognitive Maps in Rats and Man." *Behavior and Psychological Man*. University of California Press, 1951.

[24] Tulving, E. "Episodic and Semantic Memory." *Organization of Memory*, Academic Press, 1972.

[25] Widrow, B. and Hoff, M.E., "Adaptive Switching Circuits." *1960 WESCON Convention Record Part IV*, 1960, pp. 96-104.

[26] Wood, E.R., Dudchenko, P.A., Robitsek, R.J., and Eichenbaum, H.J. "Hippocampal Neurons Encode Information about Different Types of Memory Episodes Occurring in the Same Location." *Neuron*, Vol. 27, No. 3, 2000, pp. 623-633.