Categorization of Dynamic Environments with CBIM

CS81: Adaptive Robotics, Spring 2012

Taylor Nation

1 Abstract

Category Based Intrinsic Motivation is a class of Intrinsic Adaptive Curiosity that uses a Growing Neural Gas to categorize a robot's environment based on its sensory inputs. It has been proven to be an effective means for categorization, but only in a restricted domain. The purpose of this experiment is use CBIM to train a robot with both translation and rotational ability in the Cartesian plane. I will then analyze the results to determine if CBIM is an appropriate method for categorization in a very dynamic environment.

2 GNG, CBIM, and IAC

Category Based Intrinsic Motivation (CBIM) is a class of Intrinsic Adaptive Curiosity (IAC) which uses a Growing Neural Gas to categorize a robot's environment based on sensory inputs. For ease of explanation, in the following sections, IAC and GNG will be described. In the third, CBIM will be described in terms of its IAC and GNG components.

2.1 *Growing Neural Gas*

A Growing Neural Gas (GNG) is a self organizing map; it consists of a set of N nodes, and E edges, and adjusts itself to reduce its euclidian distance from elements of the input space. However, instead of having the fixed topology inherent to a self-organizing map, the GNG can add units to itself to better fit the space. [1] Describes the algorithm for the GNG's growth:

1. Given an input, calculate the points in the GNG that are nearest to it, denoted v_1 and,v_2, respectively.
2. For all vertex with edges coming from v_1:
   (a) Add 1 to age.
   (b) Add $||distance(input, v1)||^2$ to a counter variable
   (c) Move v_1 closer to the input by a fraction of that counter variable.
3. For v_2 to v_n
   (a) Move v_i by a specified fraction of distance(input, v_i)

4. Reset the age of the edge between v_1 and v_2 to 0, or create it, if it doesn't exist.

5. Remove any points that are older than the maximum specified age, and points with no edges going out from them.

By using this algorithm, a GNG can grow over time to fit the topology of any input space. In the context of an adaptive environment, the GNG can also mold existing nodes to fit changes in a pre-existing input space.

2.2 *IAC*

Intrinsic Adaptive Curiosity (IAC) is an algorithm that simulates curiosity in a robot by driving it toward situations that provide the maximum amount of learning from a situation. As described by Pierre Yves-Oudeyer in [2], the first component of IAC is the robot's learning machine, which produces a prediction about the sensory state at time $t+1$ based on the current sensory input at time t, denoted $S(t)$ and its actions $A(t)$. Once the predictor has made its guess about $S(t+1)$, the robot executes the action and views the actual effect that the action has on $S(t+1)$. Here, we will denote that effect $Sa(t+1)$. The error between $S(t+t)$ and $Sa(t+1)$ is then calculated, and used to calculate the mean error for the last n time steps (n is a specified parameter):

$$MeanError(t) = \frac{\sum_{i=0}^{i=n} E(t) - E(i)}{n}$$

IAC tracks the mean error for a specified number of time steps, to see if it is decreasing or not. Finally, the learning progress is calculated from the difference between the error at time t and the time $t-n$. At time t, the robot's learning progress is calculated:

$$Progress = -(MeanError(t) - MeanError(t-n))$$

The idea behind IAC is to keep this learning progress as high as possible, as it indicates an overall reduction in the error of the robot's predictions, which is learning, by definition. However, Oudeyer presents an interesting issue with that definition of learning progress: since the progress is just calculated from error over time, a robot can do some unpredictable task for a while, and then switch to a very predictable task, and by IAC's calculation of progress, it's learning. In reality, though, that is clearly not the case.

In order to circumvent this issue, IAC employs an "expert", or prediction machine specific to each situation. It keeps track of every example that the robot experiences, from time steps 0 to t. At the onset, IAC only contains 1 expert, and then splits into more when the number of examples that the expert contains exceeds a given threshold. Once this split occurs, there is a second parameter, called a cutting value, which determines the differences between 2 experts. It splits one of the dimensions of the sensorimotor space in half; if the value is above the cutting value, it goes into expert e_1, if it's below the cutting value, the experience goes to expert e_2. Given all of these machines and the way that they work, the end result is that the that a robot will select an action from a list of candidate choices $(c_1, c_2, ..., c_n)$, predict what effect the action will have. The robot then executes the selected action, and IAC uses the calculated error to get learning progress at that time step. Over time, the robot strives to reduce error (thus maximizing learning progress), and so it focuses on elements and tasks that are predictable. As these tasks become more and more predictable, learning progress decreases, and the robot can move on to more complex elements of the environment.

2.3 *Category Based Intrinsic Motivation*

Category Based Intrinsic Motivation (CBIM) is a hybridization of the GNG and IAC developmental structures. CBIM uses IAC to generate curiosity, and uses a GNG to conform to the inputs that it is given. In order to store all of the robot's past n similar experiences being stored in an expert like they are in IAC, the experts in CBIM are nodes in the GNG. Each node has a model vector, which is an abstraction of a specific type of sensory experience. When the robot receives the sensory vector $S(t)$, it compares that vector to all of the other model vectors in the GNG. If $S(t)$ and the model vectors are similar enough, then the model is fed into that node of the GNG, which also contains a neural network (for the prediction machine). If no model vector is similar enough, the GNG will create a new node. In that way, every set of sensory inputs that the robot receives is represented in the network. Once an appropriate region of the GNG is found, the region expert generates a list of candidate actions, some random, and some based on previous time steps. An action $A(t)$ is selected from the list, and added to the sensory vector to create the sensorimotor vector SM(t).

$$S(t) + A(t) = SM(t)$$

$SM(t)$ is fed into the neural network, which generates a prediction of the sensory input in the next step, $S(t+1)$. In this way, the robot attempts to predict the effect that its actions will have on the environment. CBIM calculates error in the same way that IAC does, and strives to minimize error and increase learning progress as well. In conjunction with the GNG's ability to map itself to the inputs received, CBIM allows a robot to categorize and generalize its environment. CBIM makes for a better artificial curiosity method than plain IAC, mainly due to its use of a GNG. The GNG's unique implementation allows CBIM to create experts only when it's necessary, as opposed to every n time steps. In this way, the GNG more accurately represents the sensorimotor space, whereas IAC's algorithm is a lot more wasteful. It creates experts when it doesn't need to, and keeps them around far past their usefulness. In a very dynamic environment, IAC's method of categorization would be ineffective, and computationally wasteful. CBIM's GNG-based architecture makes it a prime candidate for categorization in a dynamic environment.

3 Related work

3.1 Oudeyer and Kaplan's IAC experiment

My experiment with CBIM is influenced by work done with IAC[2,3], and is also an extension of an experiment done with CBIM in 2009[1]. In [2], Pierre Yves-Oudeyer experimented with IAC applied to a specific task. Oudeyer's experiment contains a robot and a ball in a simulated world. The robot had a 3-dimensional action space; it could move its left and right wheels (called LW and RW, respectively) in the range [-1,1], and it could emit a frequency in a range [0,1]. The ball moved based on this frequency. The frequency was divided into 3 subranges, which the robot was left to discover for itself as part of the task. The robot also had a 2-dimensional sensory space; it had infrared sensors that gave information as to the robot's distance from the ball and the walls of the environment. Put together, we have:

$$SM(t) = (LW, RW, FrequencySelection, BallDist, WallDist)$$

The random bouncing of frequency 1-[0,0.33] was the hardest for the robot to track, as the robot did not know where the ball will be; the location was not only random each time, but was also affected by the robot's wheel motor outputs. Depending on whether the robot moved forward or backward, it

could end up behind or in front of the ball. This task was completely unpredictable. Frequency 2-[0.34,0.66] made the ball stop bouncing. A stationary ball was be fairly easy to track, because when the ball stopped moving, the robot could easily predict where it was (since it did not go anywhere). However, a degree of difficulty occurred when we consider that the robot was moving forward and backward. If it moved forward, and emitted frequency 2, then the ball was be close to it. If it moved backward, the ball would be further away, and if the robot did not move, the ball will be in a middle distance. All of these variances within one frequency made it a fairly difficult, although not impossible, categorization to learn. Finally, frequency 3-[0.67,1] made the ball bounce directly onto the robot. This was the easiest frequency to categorize, because the ball was be in the same place regardless of what other motor outputs the robot had.

IAC was useful in categorizing the robot's behavior. Although at the onset of the experiment the robot had no idea of what its actions meant (other than their representation as numbers), it quickly grew to learn the three frequency output ranges and what they did. In his paper, Oudeyer shows graphs show both the mean error predictions for each frequency, and the number of times that frequency gets selected. As expected, considering the way IAC works, frequency 1 has both the highest mean error and lowest amount of times selected. This low number of selections stems from the fact that the Learning Progress was bound to be low, as the unpredictability of the situation makes the error level too high. Frequency 3 started off at a high selection level, due to its high learnability. However, as the task became more predictable, the mean error dropped drastically, and its selection dropped off as well. It coincided with the rise in selection of frequency 2, which is the optimal learning task. There was just enough unpredictability to make the robot select it at first, and then learnable enough to maintain the robot's interest.

This IAC experiment is like my own in that it gives the robot no explanation of its motor space and requires it to categorize everything on its own. In addition, the robot follows a specific developmental trajectory, which begins with easy tasks and moves onto harder ones. However, the major divide between the two of experiment lies in the fact that I use CBIM instead of IAC. Since CBIM is a class of IAC that uses a GNG for the experts, there will more than likely be a smaller number of experts in my experiment.

3.2 Lee, Walker, Meeden, and Marshall's CBIM Experiment

My experiment is an extension of the one outlined in [4], an experiment that employed CBIM in a real-world environment. The experimental environment was a circle with green walls, a motionless red robot, a blue robot that moved back and forth in a straight line, and a Rovio robot (the agent in the experiment). These elements were representative of the levels of dynamism found in the real world. The walls were static, and were present regardless of where the robot turned. The red robot was also static in the sense that it was not changing its Cartesian location. However, depending on where the robot was looking, the red robot may not have been within the field of view, a fact which created a dynamic element. Finally, the blue robot was moving back in forth, albeit in a predictable manner. This is the most dynamic element of the environment, because not only was it moving independently in the XY plane, it was also changing location relative to the robot. Depending on whether or not the Rovio got the timing correct, it was conceivable that the robot would never be able to track the blue robot's movement pattern.

For sensors, the Rovio was equipped with a camera that contained 4 filters: red, blue, green, and blobify. The red, blue, and green filters return a binary result; they only activate when the relevant color is present in the environment. The blobify filter draws a rectangle around an object of the selected color and returns the x and y coordinates of the endpoints of the rectangle. These coordinates were used to coordinate the size/area of the blob. The Rovio was also given an artificial sensor that gave the position of the blob relative to the robot; it returned 0 for left, 1 for directly in front, and 2 for right. This creates the 5 dimensional sensory space:

$$S(t) = (Red, Blue, Green, BlobArea, BlobPosition)$$

In the motor space, the robot had the ability to turn from 0 to 180 degrees, a value normalized to the range [0,1], with 0 being all the way left and 1 being all the way to the right. The Rovio could also select the color that it wanted the blobify filter to focus on. The resulting 7 dimensional sensorimotor space is:

$$SM(t) = (R, B, G, BlobArea, BlobPosition, TurnValue, ChannelSelection)$$

The robot did not have a set task for this experiment; it had complete autonomy within its sensorimotor space, and could explore as it desired. The

results focused on the growth of the GNG, which reflects on the robot's sensory experiences over time. At the onset of the experiment, the GNG grew quickly, as pretty much everything the robot had seen was so new that it generated enough error to make a new GNG unit. As time progressed, however, the GNG's growth slowed as the environment became more and more familiar due to successes in categorization. The data also showed that the Rovio followed a developmental trajectory. The robot first focused on the static green walls, but it slowly began to be able to track the red and blue robots accurately as the experiment progressed. This suggests that the Rovio learned the robots' motion patterns, and could correctly predict where they would be, given the proper set of circumstances.

4 Motivation

The motivation for this project is to prove the utility of CBIM in a real world environment. CBIM has a lot of potential to be an acceptable algorithm, mainly due to how it categorizes information. The use of a GNG, which grows specifically to fit the sensory space, seems to accurately and efficiently mimic the way humans learn new tasks. In the same way that the GNG adds units to fit an unfamiliar experience, people pick up new techniques to navigate situations with which they are unfamiliar. Using a GNG is also very efficient, as it grows only to fit the environment, instead of adding units every n time steps, like IAC does. However, the only testing of CBIM currently completed only exists in a limited environment, a space that only allowed 180 degrees of rotational freedom, and no ability to move forward or backward. The drawback to categorizing in such a limited environment is that although the robot can understand some properties of the objects in the world, the robot cannot understand the effects of its own movement on the items. For example, in [4], the robot eventually learned to predict where the red robot would be if it rotated. Making the accurate prediction as to the location of the red robot is an important step, but what about the effect that moving closer to the red robot would have? Using the categorizations gathered in [4], the robot would not be able to accurately predict the outcome of moving closer (the red would slowly grow to take up a greater percentage of the acting robot's field of view, while other colors would take up an increasingly smaller part). This could prove problematic in a fluid, realistic environment.

While CBIM worked well in the limited-environment, it is important

that it also work well in a space where the robot can navigate the entire 2-dimensional plane. A plane more closely simulates a real world environment, where the robot will have to move in an unspecified number of orientations to complete a task. It is the sentiment of the author that once the robot can learn and categorize elements in 2-(and eventually 3-) dimensional space, it will possess an adequate understanding of its effect on its surroundings. Such an understanding is the necessary basis for the ability to perform other tasks in a real-world, dynamic atmosphere. The environment that the robot has to categorize in this experiment fits this motivation well, as it contains the three levels of dynamics that humans (and as a result, the robot) will encounter in everyday life. The robot's success or failure in this environment will be indicative of CBIM's appropriateness for the development of artificial curiosity in an ever-changing landscape.

5 Experimental Setup

As an extension of [4], the environmental setup of this project will be very similar to the environment described. The world will contain a Rovio robot, green walls, a static red object, and a dynamic blue object. These objects are meant to show varying levels of change, much like an agent would experience in the real world. The green walls will be within the robot's view most of the time, mimicking a very static area. The red object will be in a fixed location, so it may or may not be in the robot's view, depending on its motor outputs. The blue object, the most dynamic, will take up a minuscule part of the environment. Its motion path is A) determined largely by the robot and B) fairly unpredictable, so it will be hard to categorize.

The biggest difference between this experiment and the one in [4] is that this project takes place in a simulated environment in Pyrobot. Since CBIM categorizes sensory experiences, it would not be desirable for the real world's ambient noise to detract from the categorization process. Pyrobot's environments are noiseless, so the data will be noise-free. In addition to working in pyrobot, The red robot was placed with a red cube in the corner of the world. This should make no difference, as the red robot was not moving at all. Its replacement with another static object will produce negligible effects.

The final difference is the blue element of the world. The blue robot has been replaced with a small puck. The change occurred because using the ball would provide a lesser, and substantially different type of motion than

that of the robot. In [4], the robot moved in a set path, regardless of the robot's intervention. Although this is definitely a part of the real world, it was felt that more elements of a realistic environment would not be moving independent of action by an agent. The blue ball moves when the robot makes contact with it; ideally, the Rovio will be able to see its effect on the ball and make a categorization to this effect.

In the sensory space, the robot has the same 5-dimensional vector as described in [4]. The Rovio will register activity on the Red, Blue, and Green color channels, which are binary outputs. In addition, the blobify filter will return the area of the biggest blob on the selected channel. The Rovio also possesses the artificial sensor that gives the position of the blob relative to the robot. In this experiment:

$$S(t) = (RedActive, BlueActive, GreenActive, BlobArea, BlobPosition)$$

The motor space is what differentiates this experiment from that in [4]. As the next step towards proving the usefulness of CBIM in a setting with real-world variability, the robot was given translational ability in addition to rotational ability. The translational motor output was in a range from [0,1], with 0 being no motion, and 1 being full speed ahead. The rotation is in a range from [-1,1], with -1 being a hard left and 1 being a hard right. The Rovio was not given the ability to move directly backward, as this resulted in an unnecessary choppiness of motion. However, it could achieve a net backward motion through a combination of turning around and moving forward. Also, in keeping with the realistic action space, the robot was also given a gripper with which it could interact with the components of the environment. The robot retained its ability to select the color it chose to focus on, as well. The motor space looked like this:

$$M(t) = (Translation, Rotation, ColorChannelSelection, GripperOpen/Close)$$

When combined, the robot's full sensorimotor vector had 9 dimensions, and looked like this:

$$SM(t) - (R, G, B, BlobArea, BlobPosition, Translate, Rotate, ColorChannel, Gripper)$$

Given a brain that used CBIM to categorize the world, the robot was set loose on the environment for 25,000 time steps, with no goal other than to learn autonomously. The experiment was run 5 times.

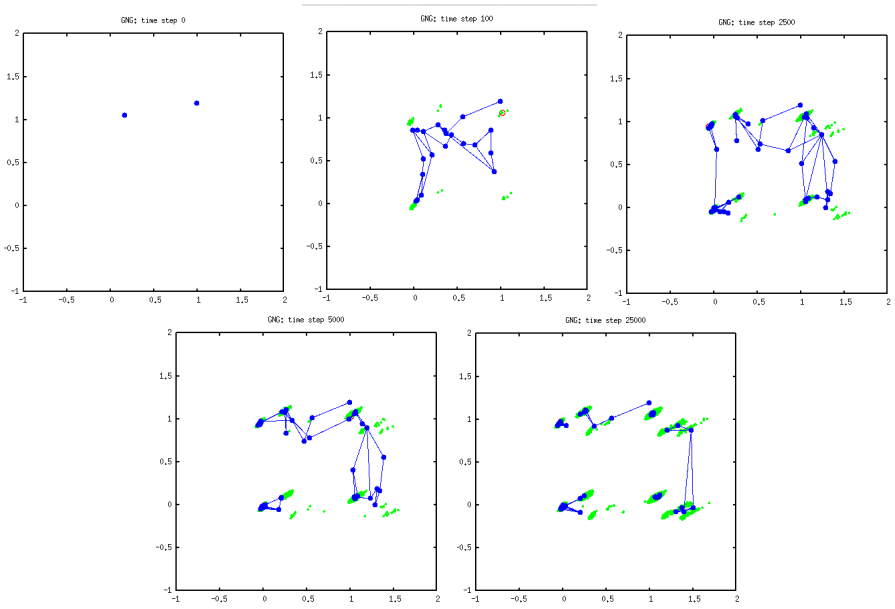

Figure 1: The GNG grows rapidly at first, then slows down and self-orgaizes to fit the input space.

6 Analysis of results

6.1 Number of Units Grown

The number of GNG units grew, as expected. As shown in figure 1, after starting with just 2 nodes in time step 0, the GNG first grew rapidly as it encountered unfamiliar elements of the input space. As the experiment continued, the Rovio stopped running into new areas and the GNG stopped growing. The GNG also showed some clustering in certain areas of the map, which is clarified in the 3D of the GNG shown in figure 2. Based on the workings of the GNG, this suggests that the dynamic environment was producing many inputs in the same sensory neighborhood (e.g. multiple instances of being near the green wall or red cube). All of these similar inputs are a result of having an environment with so many degrees of freedom; although the sensory space is different enough to warrant different GNG units, some of the nodes cluster together due to overarching similarities in the visual inputs. This shows the variances within subsections of a dynamic environment, which is created by uninhibited movement along the X and Y axes.

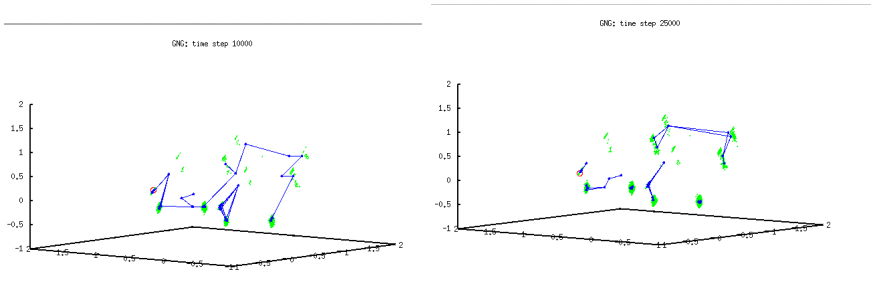

Figure 2: A 3D rendering of the GNG. The clustering shown in the 2D map is depicted a bit more clearly here. The clusters represent diferences within the green, red, and blue subspaces of the map.

6.2 Decrease in error over time

Driven by IAC, CBIM strives to reduce error in all of its experts over time. As a result, I expected the simpler elements of the environment (in this case, the green walls) to see error reduced as CBIM adjusted the nodes in the GNG to accurately fit the sensory space. Conversely, more complex elements of the environment (like the red cube and blue ball) would keep a higher level of error, though it should decrease over time. To look at an example of the effects that freedom in the XY Plane had on the green channel, we look at CBIM's GNG unit 45:

$$[1, 1, 0, .527, 0, .382, .811, .990, 0]$$

This vector is indicative of a situation in which the Rovio was near the wall with the green blobify filter selected. The vector indicates that the Rovio in front of a wall, as the green blob was directly ahead of it. Since the blob took up almost half of the robot's point of view, it is safe to assert that the Rovio was fairly close to the wall. In time step 1911, the prediction was that if the robot turned left, the wall would be in front of it and take up about half of its screen still. This prediction proved to be correct. As a result, the unit error was 0.01759, which is very low. However, if we view the graph of the error over time, some spikes in the error becomes visible. This is because since green takes up such a large amount of the sensory space, there are many different instances where the robot will be near something green. The robot experiences many types of "green experiences" and as it moves about the world, the GNG has to readjust to stay close to it. A side effect

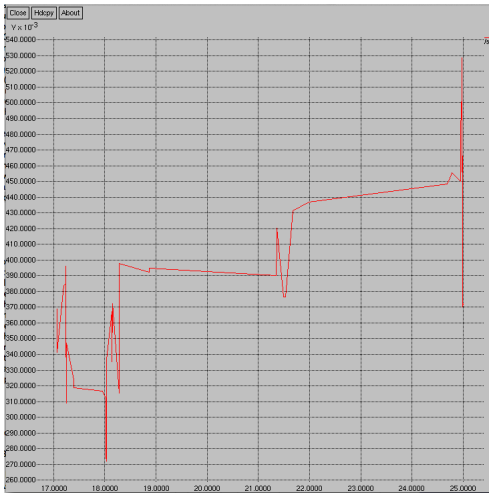

Figure 3: Error in region 45 spikes as the GNG gets too far from the input space, then decreases as the map readjusts itself.

of this is the increase of error; if the space is changing quickly, the GNG will generate a lot of error before it moves to a nearer location or creates a new node. The spikes in error show that the GNG is keeping up with the sample space as it changes over time.

The error levels in the red and blue channels are lower, which supports the idea of the GNG growing to fit the space. As previously stated, the red elements of the environment take up a smaller portion of it, and do not move. As a result, the amount of situations in which the robot has to make predictions about red are smaller. That may make it easier for CBIM to predict the outcomes of red, as there is a smaller set of possible outcomes. The GNG also only has one small area in the space to fit. This is reflected in the error in GNG units that indicate a red color selection. Take unit 34:

$$ModelVector : [0, 0.996, 0, 0.021, 0.0070.737, 0.414, 0.268, 0.032]$$

This is indicative of a situation in which the Rovio is facing the wall, with the red cube off to its side and the red color active. It must be far away, as there is so little of the cube in the Rovio's line of sight that it doesn't activate the red filter on the camera. Given the sensorimotor vector

$$SM(t) = [0, 1, 0, 0, 0, 0.938, 0.448, 0.077, 0]$$

CBIM predicted that if the Rovio would not be able to see the red cube if it made a right turn away from where it currently was while moving forward. In this case, it was correct, as CBIM had begun to understand the effect that the Rovio's actions had on the environment.

The blue channel did not quite meet expectations for the experiment, largely because the ball proved to be a less dynamic element than expected. Although the ball is small enough that it is not consistently in the field of view, it may have also been so small that the Rovio did not notice it. In most of the GNG units, the blue color is not even active. Additionally, since the ball only moves when it is touched, the Rovio had no way of knowing that the ball was any more movable than the red cube. The third issue was that if the Rovio did hit the ball by mistake, the path that the ball would take was more than likely too complex for the robot's brain to calculate. For these three reasons, the GNG does not have very many units that reflect interaction with the blue ball.

6.3 Color Channel Selection

Figure 4 shows the color selections over time. As discussed previously, the blue ball was either too unnoticeable or too unpredictable, and its selection frequency is substantially lower than that of the red or green channels. Looking at the green channel, over the first half of the experiment, it rises to be the most selected choice, peaking at almost 50

It is important to note, however, that the green channel selection does not drop off completely, and the oscillation between red and green channel selection is drastic. That may stem from the fact that since the Rovio has so much freedom in the XY plane, the environment is too complex to categorize completely in such a short time period. The mean error over time did not reduce enough for the green channel to create a high enough learning progress for CBIM to ignore it completely. In the same vein, although red's selection appeared to be on an upward trend since the 15000 time step, it was not consistently the most selected channel. Again, due to the amount of freedom that the Rovio had, CBIM did not have the necessary time to consistently select the red channel due to the fact that it was still dealing with the green one.

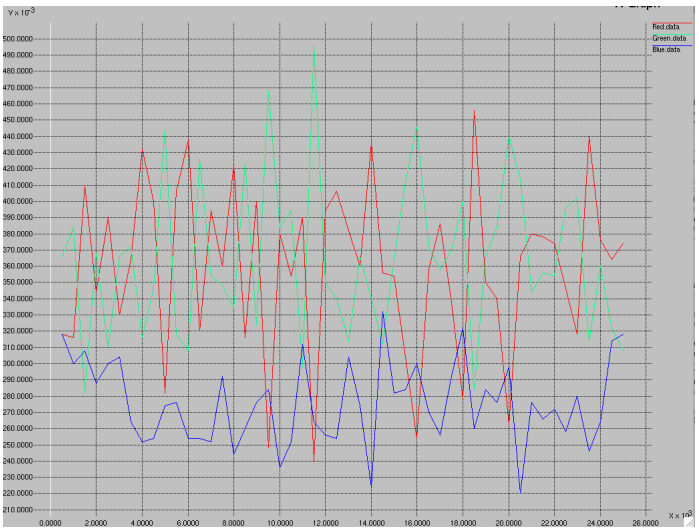

Figure 4: Red, Blue, Green color channel selection over time.

7 Conclusions and Future Directions

From the results taken from the experiment, I believe that CBIM can be an effective tool for categorization in a dynamic environment, but it can take a lot longer than previously expected. The results in 6.1 show that the GNG will grow to fit its sensory space properly, a task that is essential for categorization and autonomous learning to take place. However, the results from sections 6.2 and 6.3 indicate that more time is necessary to appropriately categorize the environment. I believe that is because when the environment has so many variables, there are too many permutations of the sensorimotor vector to categorize in a short period of time. We can see this from the graphs in figures 1 and 2 , which indicate clusters of nodes in the same area. That is representative of different situations associated with one color, a result of having an environment that changed so much. However, some of the trends seen in GNG growth, decrease in error over time, and the recency of channel selection indicate that given more time CBIM would be an acceptable method for autonomous learning in a dynamic environment.

For future directions of this project, possible revisions of the experimental setup include making the ball bigger and placing it directly in front of the Rovio. Such a revision would force the robot to interact with the ball

every time step, so that it eventually becomes an explorable element of the world. Another would be to replace the ball with a robot that moves, like the experiment in [4]. That would more elucidate CBIM's reactions to overly complex tasks/items, as this experiment did not successfully simulate that type of an experience. A third option would be to extend this project into a 3-dimensional world, which would most directly simulate a real-world environment. In time, I hope that a robot trained in CBIM would be able to apply that categorization to perform a task in a separate world, illustrating the applicability of categories formed with CBIM. In conclusion, CBIM appears to be an appropriate categorization algorithm, but it needs a longer training period to be useful in a dynamic environment.

Bibliography

[1] *A growing neural gas learns topologies*. Advances in Neural Information Processing Systems 7, Bernd Fritzke (1995)

[2] Oudeyer P-Y, Kaplan , F. and Hafner, V. (2007) *Intrinsic Motivation Systems for Autonomous Mental Development*, IEEE Transactions on Evolutionary Computation, 11(2), pp. 265–286. DOI: 10.1109/TEVC.2006.890271

[3] Baranes, A., Oudeyer, P-Y. (2009). *R-IAC: Robust intrinsically motivated exploration and active learning*, IEEE Transactions on Autonomous Mental Development, 1(3), pp. 155-169.

[4] *Category-based intrinsic motivation*, Proceedings of the Ninth International Conference on Epigenetic Robotics, Rachel Lee, Ryan Walker, Lisa Meeden, and James Marshall (2009)