Curiosity-Based Learning Algorithm for Interactive Art Sculptures

by

Matthew Tsz Kiu Chan

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Electrical and Computer Engineering

Waterloo, Ontario, Canada, 2016

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

Combining techniques in architecture, the arts, electronics, and software, the Living Architecture System Group (LASG) develops interactive art sculptures that engage occupants in an immersive environment. The overarching theme of this project is to develop architectural systems that possess life-like qualities. Recent advances in miniaturizations of computing and sensing units enable system-wide responsive behaviours. Though complexity may emerge through superpositions of a set of simple and prescripted behaviours, the responses of the system to the occupants remain rather robotic and ultimately dictated by the wills of its designers. In this thesis, a new series of sculptural system was initiated, implementing an addition layer of behavioural autonomy.

The sculpture's abilities to automatically generate interactive behaviours and adapt to changes are actualized by the introduction of the Curiosity-Based Learning Algorithm (CBLA). It is an reinforcement learning algorithm which selects actions that lead to maximum potential knowledge gains. It allows the sculptural system to construct models of its own mechanisms and its surroundings through self-experimentations and interactions with human occupants. A novel formulation using multiple learning agents, with each assuming a subset of the system, was developed to integrate a large number of sensors and actuators. They form a network of independent, asynchronous CBLA Nodes that share information about localized events through shared sensors and virtual inputs. Given different network configurations of the CBLA system, the emergence of system behaviours with varying activation patterns was observed.

To realize the CBLA system on a physical interactive art sculpture, an overhaul of the previous series' Arduino-based interactive control system was necessary. CBLA requires the system to be able to sense the consequences of its own actions and its surrounding at a much higher resolution and frequency. This translates to the need of interfacing and collecting samples from a substantially larger number of sensors. In that respect, the previous system is inadequate in terms of computing power, data transfer rate, and the number of interfaces. Therefore, a new series of hardware as well as control system software was developed. This enables the control and sampling of hundreds of devices on a centralized computer through USB connections. Moving the computations from an embedded platform simplifies the implementation of the CBLA system, which is a computationally intensive and complex program. In addition, the large amount of data generated by the system can now be recorded without sacrificing response time nor resolution.

An experimental test bed was built to validate the behaviours of the CBLA system. It is small-scale interactive art sculpture resembling previous sculptures that were displayed

publicly by the Philip Beesley Architect Inc. (PBAI). Experiments were done to demonstrate the exploratory patterns of CBLA as well as the collective learning behaviours emanated from the CBLA system. Furthermore, a formal user study was conducted to better the understanding of users' response to this new form of interactive behaviours. Comparing with prescripted behaviours that were explicitly programmed, the participants of the study did not find this implementation of the CBLA system more interesting. However, the positive correlations between activation level, responsiveness, and users' interest levels revealed insights about users' preferences and perceptions of the system. In addition, observations during the trials and the responses from the questionnaires showed a wide varieties of users' behaviours and expectations. This suggests that, in future work, results should be categorized to analyze how different types of users respond to the sculpture. Moreover, the experiments should also be designed to better reflect the actual use cases of the sculpture in order to validate if high level of activations and responsiveness can indeed engage the users over a longer period of time.

Acknowledgements

I would like to thank all the people who made this possible.

Dedication

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction [1]

Interactive arts are a type of art form that requires the involvement of the spectators to achieve its purpose. With recent advances in capabilities and miniaturization of computers, sensors and actuators, artists now have access to more tools to create highly complex interactive artworks. In the Hylozoic Series of kinetic sculptures built by Philip Beesley Architect Inc.(PBAI)[2], the designers use a network of microcontrollers to control and sample a sizable number of actuators and sensors [3] [4]. Each node in the network can perform a simple set of interactive behaviours. While the behaviours have previously been either prescripted or random, complex group behaviours have been seen to emerge through the communication among nodes and interaction with the spectators [5].

In this work, we wanted to explore the physical manifestation of a learning algorithm in an abstract interactive art sculpture. Working with PBAI, a new series of interactive sculptures was designed and built to generate its own behaviours through learning and interaction with its physical environment. We observed how occupants of the space responds and perceive the behaviours of the sculpture.

Oudeyer et al. developed the Intelligent Adaptive Curiosity (IAC) [6], a reinforcement learning algorithm with the objective of maximizing learning progress. In his experiments, he showed that an agent would tend to explore state-space that is neither too predictable nor too random, mimicking the intrinsic human drive of curiosity, which continually tries to explore areas that have the highest potential for learning new knowledge. The learning mechanism developed for this thesis built on Oudeyer's IAC and applied it on an

[1] Part of this chapter is adapted from papers submitted to IROS 2015 [1] and Next Generation Building [2]

[2] Philip Beesley Architect Inc.: philipbeesleyarchitect.com

architectural-scale distributed interactive sculptural system. We called it Curiosity-Based Learning Algorithm (CBLA). A CBLA system is comprised of a network of asynchronous learning agents that are linked virtually and physically.

The CBLA functions by exploiting the system's inherent curiosity to learn about itself, much like an infant might learn by exercising groups of muscles and observing the response. In its simplest form, the algorithm chooses an action from its action repertoire to perform, and measures the response. At the same time, it generates a prediction of what it thinks should happen. If the prediction matches the measured response, it has learnt that part of its sensorimotor space and that space becomes less interesting for future actions. If the prediction fails to match the measured response, it remains curious about that part of its "self", as it obviously still has more to learn. It will create a new prediction and try again. This learning architecture allows the system to learn both about itself, and also about interactions with occupants, whose movements and actions create new and "surprising" responses, activating the system's curiosity.

A new series of electronic hardware and communication architecture was motivated by the introduction of CBLA. To accommodate this algorithm, it requires the system to be able to sense the consequences of its actions, similar to the human capability for proprioception. Similar to human proprioceptors, these sensors allow the sculpture to both detect its own actions, and the actions of occupants on its embodiment. For example, an accelerometer on a Fin senses both when the Fin actuator is activated, and also when the Fin is touched by a visitor during interaction. Without proprioceptors, the sculpture can only estimate its own dynamics based on a feed-forward model. For a human being, this capability is implemented through a neural mechanism known as efferent copy [7]. For example, human eyes are constantly moving while a stable image is reconstructed using the efferent copy. However, the efferent copy can be deceiving when the external environment is disturbed to conflict with the predicted model. For instance, a stationary image will appear to be moving when the eye is pressed [8]. However, if the disturbance is permanent, over time the efferent copy will be updated to reflect the new conditions, and an accurate model is once again available for prediction. Hence, not only is an accurate model of an agent's own dynamics difficult to obtain; such a model might change over the life of the installation due to wear and tear and interaction with its surrounding environments. Proprioceptors play an important role in giving the sculpture information about its own state to enable model learning and adaptation.

To better understand users' response and perception of this learning behaviours, it is important to be able to capture their responses as they are interacting with the sculpture.

1.1 Motivation

We introduced the Curiosity-Based Learning Algorithm (CBLA) to replace pre-scripted responses in the Hylozoic series installation. The CBLA re-casts the interactive sculpture as a set of agents driven by an intrinsic desire to learn. Presented with a set of input and output variables that it can observe and control, each agent tries to understand its own mechanisms, its surrounding environment, and the occupants, by learning models relating its inputs and outputs. We hypothesize that the occupants will find the behaviours which emerge during CBLA-based control to be interesting, more life-like, and less robotic.

This approach reduces the reliance on humans to manually design interesting and life-like behaviours. In systems with large numbers of sensors and actuators, programming a complex set of carefully choreographed behaviours is complicated and requires lengthy implementation and testing processes. Furthermore, this approach allows the sculpture to adapt and change its behaviour over time. This is especially interesting for permanent installations in which the same users may interact with the sculpture over an extended period of time.

1.2 Projects Components

Moreover, compared to the previous generations of interactive art sculptures produced by PBAI, the CBLA requires the control and sensing of a much larger number of actuators and sensors at a relatively high frequency. Therefore, a new series of electronic platform and control architecture that can communicate and synchronize with a distributed set of sculptural units was developed.

Additional abstraction layer was created to allow quick implementations of many different configurations. By studying the behaviours emerged and the response of the occupants, we began to understand the user's impressions perceptions of learning behaviours, and relationship between the intensity of activation and user's interest levels.

This project can be broken down into five main components that need to be solved.

1.2.1 Control hardware and software

Compared to the previous generations of interactive art sculptures produced by PBAI, the CBLA requires the control and sensing of a much larger number of actuators and sensors

at a relatively high frequency. To accommodate that, a new hardware platform was built. This hardware platform uses Teensy 3.1 device to interface with the sensors and actuators. A set of PCB was custom designed to enable control and sampling of over 24 actuators and 18 sensors. In addition, a communication protocol was developed to interface each Teensy device with a computer through USB. The computer is expected to control over 16 Teensy devices at the same time.

1.2.2 Interfacing software

A Python module was written to enable the communication with many Teensy devices simultaneously. A thread is created for each Teensy device and synchronize the copies of the hash tables on the Teensy device and the computer continuously. Applications on the computer side can control and sample the states of the Teensy devices by modifying the values in the hash tables.

1.2.3 Abstract nodes layer

An additional layer of abstraction was built to represent the sculpture as a network of node. Each node runs as its own thread and possesses a set of input and output variables. Behaviour of each node can be programmed individually. A pre-scripted version of the behaviours will be programmed to compare with the CBLA version in user studies.

1.2.4 Curiosity-Based Learning Algorithm

The CBLA is a type of reinforcement learning algorithm with the reduction of prediction error as the reward. During the learning process, it will explore regions of the state space that are neither too predictable nor too random; it focuses on areas that have the highest potential for new knowledge. To structure the learning process and identify interesting regions of the state-space, the CBLA automatically segments the state-space into regions; an expert in each region makes predictions about the effects of an action and adjusts its prediction model based on the actual resultant state. The value of each expert is determined by its record of error reduction. This value will then determine the execution likelihood of the action associated with this expert.

1.2.5 User Study

A user study was conducted to test the efficacy of increasing users interest level in an interactive art sculpture, by using a curiosity-based learning algorithm (CBLA) to adjust the sculptures dynamic behaviours. Simply put, we would like to test whether behaviours generated using the CBLA are more interesting than pre-programmed behaviours designed by human experts. The test subjects will report their level of interest at several points in time as they interact with sculpture, with the two versions of behaviours. Afterwards, a short survey will be given to assess the subjects overall experience. The results of this study will enable designers to design more engaging and interesting interactive art sculptures.

1.3 Contributions

First, this thesis presented an adaptation of learning algorithm which was previously used on developmental robotics in an interactive arts. In [6], the state space of the robot was relatively small. However, in our case, there is an order of magnitude more input and output variables. In order to make learning such a large distributed system manageable, multiple learning agent may run simultaneously, each capturing a specific set of sensors and actuators.

Second, unlike many reinforcement learning problems, acquiring a model of the learning targets as quickly and accurately as possible isn't our main objective. Instead we want to study the learning process itself and study how curiosity affects the way that the system explores the state-space. The behaviour of the interactive art sculptures is the physical manifestation of the learning process.

Third, the user study conducted allows us to examine how human occupants interacts and perceive the action of the sculpture. From there, we can begin to understand what interests people and how different activation patterns might lead to different reactions.

1.4 Organization

In Chapter 2, related work in interactive arts, artificial life, developmental robotics, and user experience would be examined and discussed. A new electronic control system was designed and built from ground up in order to incorporate the large number of sensors and actuators and to run the CBLA. The design and implementation of hardware and

software of this system are presented in Chapter 4. Then, in Chapter 3, the working of our implementation of the Curiosity-Based Learning algorithm would be thoroughly discussed. We observed the behaviours of the sculptural system running CBLA an collected user experience survey in some of those experiments. The methodologies and findings of the experiments are presented in Chapter 5.

Chapter 2

Related Work

2.1 Interactive Arts

Interactive arts can be categorized as Static, Dynamic-Passive, Dynamic-Interactive, and Dynamic-Interactive (varying) based on the degree of the interaction between the art works and the viewers [9]. Dynamic-Interactive systems give the human viewers an active role in defining the behaviours of the system. This category introduces an agent that modifies the specifications of the art object. This additional unpredictability introduces a new dimension of complexity to the behaviours of the system. In [10], Drummond examined the conceptual elements of interactive musical arts. For an interactive system to move beyond being simply a complex musical instrument with a direct reactive mapping from inputs to generation of sound, it must possess some level of autonomy in the composition and generation of music. In addition, the interactivity should be bilateral; the performer influences the music and the music influences the performer. These concepts can easily be extended to visual and kinetic arts. Visual-based interactive systems such as the Iamascope in Beta_space [11] and audio-based systems such as Variations [12] engaged the participants by allowing them to directly affect the output of the system. Works in interactive architecture [1] [8] try to provide a responsive and immersive environment where the viewers can feel as if they belong to the system. However, most of these works are the non-varying type of Dynamic-Interactive system, as their responsive behaviours do not change. Over a longer term, the system will become more predictable and its effectiveness in engaging the users will consequently decrease. In this work, we aim to create a varying interactive artwork by emulating the characteristics of living organisms such as curiosity and learning [5].

2.2 Artificial Life

To emulate life-like behaviours, one can start by observing how human beings behave. [13], [14] modelled how human beings convey or mask their intentions through movement and applied these models on a humanoid robot. Similarly, [15] focuses on making the robot's motion more understandable by emulating the coordinated effects of human joints. Those studies focus their attention on making the intent of the robots clear. In contrast, our objective is to make robots more engaging and life-like, where unpredictability might be a desirable quality. For instance, [14] showed that the robot's perceived intelligence increased when the participants believed that the robot was intentionally deceptive. Our work investigates whether unpredictable behaviours emerging from the learning process will appear more life-like and engaging. One of the open questions in artificial life research is whether we can demonstrate the emergence of intelligence and mind [16], examined in projects such as the Petting Zoo by Minimaforms [17] and Mind Time Machine [18]. The idea of emergence of structure and consciousness is explored in many previous works in the field of developmental robotics [19] [20] [21]. Oudeyer et al. developed a learning mechanism called Intelligent Adaptive Curiosity (IAC) [6], a reinforcement learning algorithm with the objective of maximizing learning progress. In his experiments, he showed that an agent would tend to explore state-space that is neither too predictable nor too random, mimicking the intrinsic human drive of curiosity, which continually tries to explore areas that have the highest potential for learning new knowledge. However, previous work did not cover how the IAC might be scaled to a distributed system with a large sensorimotor space.

2.3 Developmental Robotics

2.4 Machine Learning

2.5 Intrinsic Adaptive Curiosity[1]

In [6], Oudeyer's goal was to develop a robot that is capable of life-long learning. The robot makes sense of the mapping between its actuators and sensors through continuous self-experimentation, without explicit instruction from human experts. Fundamentally, the

[1] An early version of this section has been published at IROS 2015 [1]

IAC is a reinforcement learning algorithm with an objective to maximize learning progress. Learning progress is defined as the reduction in prediction error. In other words, the agent seeks to explore the region of the sensorimotor space that leads to the highest improvement in prediction error. As a result, the algorithm avoids learning in the parts of sensorimotor space that are either too random or too predictable. This formulation leads to continual change in behaviour over time, as regions of the state space that are initially interesting become less interesting as the system becomes more knowledgeable about them.

The system consists of two learning mechanisms, the Classic Machine learner (classM) and the Meta Machine learner (metaM). Based on the context (the sensors' inputs) and the action (the actuators' outputs), classM computes a prediction of the consequence, i.e., the resultant sensors' inputs at the next time step. Then, it compares the actual consequence with the prediction and modifies its model in order to reduce the error in the subsequent prediction. The Meta Machine learner (metaM) predicts the error of classM. In other words, it estimates how accurately classM is able predict the consequence. The actual prediction error is then fed back to the metaM, and metaM modifies its estimate of prediction error of the newly modified classM. This change in the prediction error is recorded at each time step in the knowledge gain assessor (KGA). The KGA then computes the expected learning progress by calculating the change in error estimation. This expected learning progress is used as the reward.

In Oudeyer's paper [6], one important feature is the idea of regional experts. Each region collects exemplars of similar sensorimotor context, and has an expert that is trained on the exemplars in the region. Exemplars are the training data for the prediction model and they are collected as the system selects actions and observes the consequences. The "features" are the sensory inputs $S(t)$, and selected actions $M(t)$ at time t; and the "labels" are the resultant sensory inputs $S(t+1)$ at time $t+1$. The regional experts constrain the estimate of learning progress within their respective sensorimotor contexts. This is important because it allows each expert to use a simpler model, as it covers only a small region of the state space.

The following are the steps for the learning algorithm:

1. Read sensory inputs $S(t)$
2. Select action $M(t)$ with the highest predicted learning progress $R(SM(t))$ based on ϵ-greedy selection policy
3. Consult expert specialized in the relevant sensorimotor context $SM(t)$ to predict the expected sensory inputs $S(t+1)'$

4. Perform action $M(t)$
5. Observe actual sensory inputs $S(t+1)$ and add to the expert's training set
6. Compute prediction error $e(t+1) = S(t+1) - S(t+1)'$
7. Compute the change in prediction error $R(SM(t)) = -[e(t+1) - e(t)]$
8. Update the learning progress for the sensorimotor context $SM(t)$ based on $R(SM(t))$
9. Repeat 1 to 9 indefinitely

2.6 User Experience Survey

Chapter 3

Curiosity-Based Learning Algorithm[1]

In this section, our approach for adapting and generalizing the IAC algorithm for implementation on a distributed kinetic sculpture is presented. This includes additional parameters, modifications to action selection approaches, and the integration of multiple independent learning agents in a distributed network.

3.1 CBLA Engine

Oudeyer [22] observed during the IAC experiments that the resulting robot behaviour had similarities to children's learning and playing behaviours. We hypothesize that this type of learning mechanism is well suited to interactive architecture systems, as the learning process itself will generate interesting behaviours and be an interesting feature of the installation to the visitors.

We now adapt the IAC algorithm [6] to implement a learning architecture on the distributed architectural system described in Chapter 4. Each system node runs its own version of the algorithm, called the CBLA Engine. A CBLA Engine is a thread that is iteratively executing the steps of the learning algorithm. It is comprised of a node, an action selector, and an expert, as illustrated in Figure 3.1

[1] An early version of this chapter has been published at IROS 2015 [1]

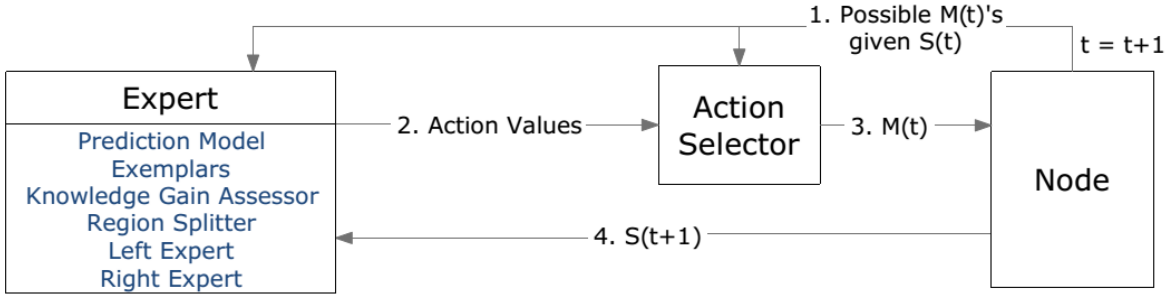


Figure 3.1: (1) At the start of each time step, the Node outputs the current possible actions $M(t)$s given $S(t)$. (2) Then, the Expert provides the Action Values associated with those possible $M(t)$ to the Action Selector. (3) After that, the Action Selector selects an action and actuates the Node. (4) After the Node has performed the action, it returns the actual resultant state $S(t+1)$ to the Expert. The Expert can then improve its internal prediction model and update the Action Value associated with $SM(t)$.

3.1.1 Node

A node represents a subset of the sculptural system. Each node is specified by set of sensor input and actuator output variables, and its own set of configuration parameters and functions, embodying the characteristics of the physical system that it represents. It communicates with the lower-level layer to set the actuator outputs and sample the relevant sensory inputs. The type and number of sensory inputs and actions are configurable, specific configurations used in our experiments will be described in more detail in Chapter 5.

3.1.2 Action Selector

The action selector selects an action based on a list of possible actions, $M(t)$, and their associated action values. It either selects an action that has the highest value (exploitation), or selects an action from the list at random (exploration). In our implementation, a version of the adaptive ϵ-greedy [23] action selection method was used. ϵ specifies the probability that a random action is selected instead of the action with highest action value. The magnitude of ϵ changes linearly based on the magnitude of the action value, q as shown in (3.1).

$$\epsilon = \begin{cases} \epsilon_{max} & \text{, if } q < q_{min} \\ \epsilon_{min} & \text{, if } q > q_{max} \\ m \cdot q + b & \text{, otherwise} \end{cases} \tag{3.1a}$$

$$m = \frac{\epsilon_{max} - \epsilon_{min}}{q_{min} - q_{max}} \tag{3.1b}$$

$$b = \epsilon_{max} - m \cdot q_{min} \tag{3.1c}$$

where ϵ is the exploration probability; ϵ_{max} and ϵmin are constants representing the maximum and minimum exploration probability; q is the action value; and q_{min} and q_{max} are constants representing the minimum and maximum action value thresholds.

When the action value, q, is low, the probability of selecting a random action, ϵ, is high since the current expected reward is low and it is worth spending time exploring new regions that may lead to higher rewards. On the other hand, when q is high, ϵ is low, since the current expected reward is high and it is worth spending time exploiting a high reward region. Limits on maximum and minimum ϵ are set to ensure that the exploration probability is kept within a range.

3.1.3 Expert

An Expert consists of a prediction model, a set of exemplars, a Knowledge Gain Assessor (KGA), a region splitter, and potentially two child experts.

An expert represents a region in the sensorimotor space defined by its exemplars. Each prediction is generated by the expert in the region with the most similar sensorimotor context. Figure 3.2 shows the internal mechanism of an Expert.

Prediction Model

The prediction model models the relationship between the node's input and output variables. At every time step, this model is used to make a prediction about the immediate future state based on the current state and the selected action. This prediction model is trained on the set of exemplars that were previously observed. These exemplars are represented as $[SM(t), S(t+1)]$ pairs. $SM(t)$ represents the sensorimotor context, i.e., the concatenation of the sensory state and the action taken at time t. $S(t+1)$ represents the observed sensory state at time $t+1$.

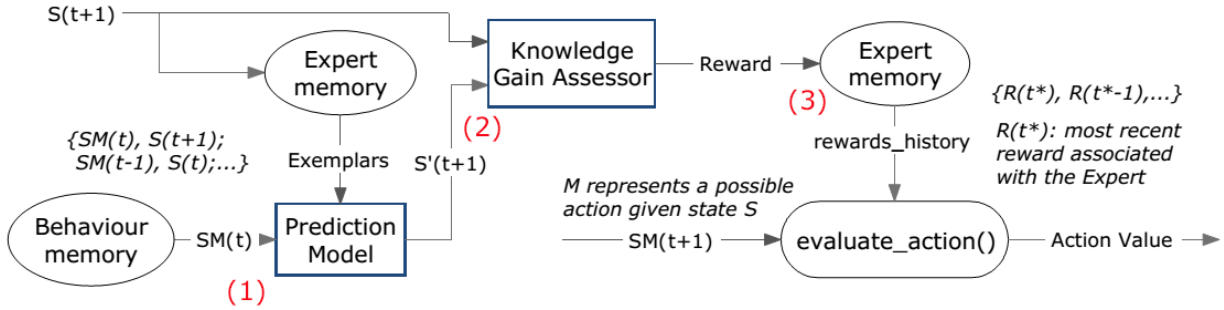


Figure 3.2: (1) The prediction model of the Expert first makes a prediction of the resultant sensor input $S(t+1)'$ based on the current sensorimotor context, $SM(t)$. (2) This prediction and the actual sensor input $S(t+1)$ are then fed into the KGA. $S(t+1)$ is also added to the collection of exemplars, which the prediction model is trained on at every time step. (3) After that, the KGA computes the reward and updates the Action Value associated with $SM(t)$ in the Expert memory. These Action Values are recalled to evaluate a possible action in the next time step the Expert is active.

In all the experiments presented in this thesis, linear least squares regression or Lasso (Least Absolute Shrinkage and Selection Operator) [24] was used. Nominally, linear regression is used due to its simplicity and the absence of parameters that require tuning. Lasso is used when the sensorimotor space includes input and output variables that are not proprioceptive. These variables tend to have low correlations with prediction model as they do not directly detect the changes in the outputs. For instance, a node with virtual inputs (explained in 3.2.2) from another node would use Lasso. Using Lasso can improve the model's predictability by reducing the coefficients associated with that are highly unpredictable. Python library *scikit-learn*'s[2],[3] implementations of the two models were used.

Knowledge Gain Assessor

The KGA calculates the reward given the predicted and actual sensory inputs. Figure 3.3 illustrates the internal workings of the KGA. The metaM in this case simply returns the average error over a window of time τ time steps ago, $< e(t+1) - \tau >$. The reward is

[2] scikit-learn LinearRegression: http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html

[3] scikit-learn Lasso: http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Lasso.html

calculated by subtracting the current mean error, $< e(t+1) >$. This reward is then used by the action selector for computing the action values.

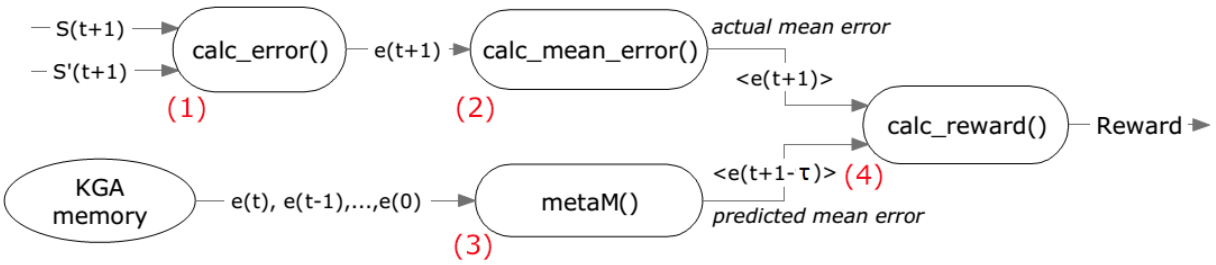


Figure 3.3: $S(t+1)$ and $S'(t+1)$ are the actual and predicted sensor input variables. (1) The KGA computes the error by taking their root-mean-square error difference. (2) After that, it computes the mean error over a window of previously calculated errors. Note that the mean error is only calculated based on errors associated with this particular region. (3) The metaM predicts the error by taking the mean error over time. (4) Finally, the KGA outputs the Reward by taking the difference between the actual mean error and predicted mean error.

Region Splitter

The system initially has only one region. All new exemplars are added to the same region. As more exemplars are added to memory, once split criteria 1 and 2 are met, a region will split into two parts.

Criterion 1:

$$N > \tau_1 \tag{3.2}$$

N is the number of exemplars; and τ_1 is the threshold that is inherited from the expert's parent.

Criterion 2:

$$e > \tau_2 \tag{3.3}$$

e is the mean error of the region and τ_2 is a threshold parameter.

Criterion 1 specifies that a region can only split into two when it contains a sufficient number of exemplars, to ensure that there will be enough training data in the sub-regions. Criterion 2 specifies that a region will only split if prediction error is high. This prevents learnt models with high accuracy from being split indefinitely. If the split criteria are

met, the Region Splitter finds a cut value and a cut dimension that minimizes the average variance in each sub-region.

After the best cut value and cut dimension are identified, the exemplars of the parent region are split between the two regions based on the cut dimensions and values. The high-level mechanism of the region splitting process is illustrated in Figure 3.4.

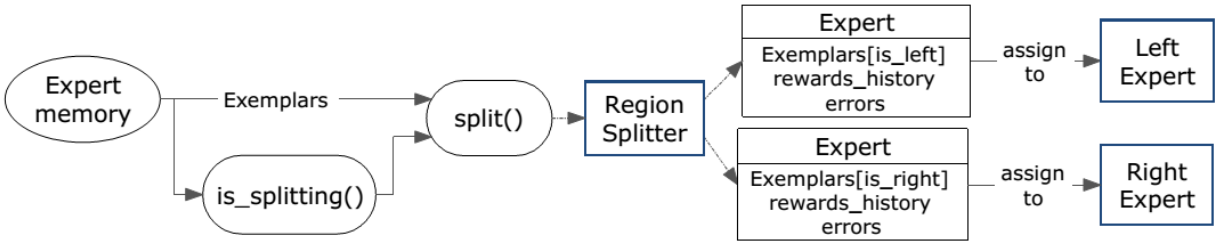


Figure 3.4: When the function split() is called, it first checks if the split criteria are met by calling "is_splitting()". If the split criteria are met, it forwards the exemplars to the Region Splitter. The Region Splitter then splits the exemplars into two and assigns them to the Left and Right Expert. Other properties such as the previous errors and rewards are passed on as well.

3.1.4 Pseudocode of CBLA

The overall process is described in the pseudocode for the CBLA and is shown as Algorithm 1.

3.1.5 Differences between the CBLA Engine and the IAC algorithm

In developing the CBLA Engine described above, several adaptations have been made to the IAC algorithm to enable application to the kinetic sculpture installation.

Region Splitter

Several improvements are made to the logic of the Region Splitter. As we run the algorithm over an extended period of time, the expert might have already learnt the model of the

Algorithm 1 Pseudocode for the CBLA

1: $t \leftarrow 0$
2: $S(t) \leftarrow S(0)$
3: **loop**
4: $[Possible\_M(t)] \leftarrow Node.get\_possible\_action(S(t))$
5: $[Action\_Value(M(t))] \leftarrow Expert.evaluate\_action(Possible\_M(t))$
6: $M(t) \leftarrow Action\_Selector.select\_action([Possible\_M(t), Action\_Value])$
7: $Node.actuate(M(t))$
8: $S(t+1) \leftarrow Node.report()$
9: $S'(t+1) \leftarrow Expert.predict(S(t), M(t))$
10: $Expert.append(S(t), M(t), S(t+1), S'(t+1))$
11: $Expert.split()$
12: $t \leftarrow t+1$
13: **end loop**

system as best as possible given the available sensory information. Since Oudeyer's method simply splits a region when the number of exemplars exceeds a certain threshold, it will continue to split regions even when the expert associated with that region has already acquired a good model with low prediction error. This is undesirable as it leads to over-fitting and makes the system harder to adapt when the system itself or the environment changes.

In our implementation, an additional prediction error threshold is added to prevent regions from splitting when the prediction error is low. Moreover, if the split does not improve the prediction accuracy, there is no reason to split the region. After the split, the split quality must also be above a threshold. If it is not, the split will be retracted. This split quality is measured by the magnitude of the average within-group variance relative to the overall variance.

During the early stages of the learning process, learning opportunities are plenty. Setting the split number threshold too high can hamper the speed of learning. However, over time, easy learning opportunities diminish and complex regions require a larger number of exemplars. Thus, the split number threshold grows at a constant rate every time a split happens. This way, regions that are less explored can maintain low thresholds which promote learning, while mature regions have higher thresholds which promote gathering of more exemplars and reduce over-fitting.

In addition, in [6], the cut value and cut dimension are determined by generating a set of random cut values for each dimension and computing the corresponding variances of the

resultant sensor inputs. The cut value and dimension pair with the lowest total variance is selected. The total variance is the sum of the variances of $S(t+1)$ in each dimension. Our implementation has made some changes to the way regions splitting is done.

First, in order to the interdependency among dimensions into account, instead of using the $S(t+1)$ value directly when computing variances, herein, its principal components are used. Using its principal components can better reflect the underlying structure of the data by emphasizing the differences among data points along the direction of maximum variance. Note that we do not compute the principal components of the $SM(t)$, where the cut value and cut dimension are specified. Instead, the evaluation is based on the principal components of $S(t+1)$ for which we want to minimize the variance.

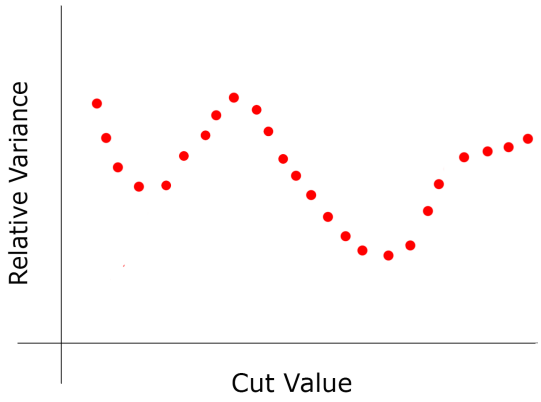
Second, if one dimension tends to have a lower variance than another, that dimension is much more likely to get selected, even if the cut does not lower its resultant variance. Since region splitting aims to separate groups into low variance regions, this is unproductive as it does not decrease the variance. Instead of using the variance, relative variance is used. It is computed by taking mean of the variance of the region divided by the total variance of the overall region before splitting. This gives a better indicator on whether or not the split results in generating more concentrated regions. A relative variance of 1 implies that the split does not improve and should not be cut in this way. On the other hand, a low relative variance implies that the split decreases the regional variances. This method can reduce the amount of unproductive splits that fragment the state space.

$$S_{relative} = \frac{S_1 + S_2}{S_{1,2}} \tag{3.4}$$

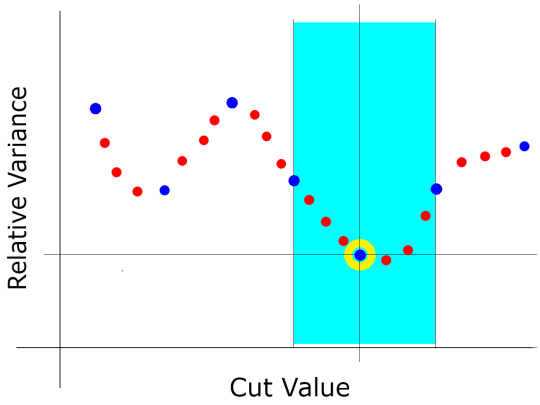
S_1 and S_2 are the total variances of the two regions split using the candidate cut value and cut dimension; $S_{1,2}$ is the total variance of the two combined regions.

Furthermore, this method is susceptible to leaving a small number of sample points near a desirable cut value since the random selection might not pick up the exemplars that are right at the edge. On the other hand, although one can simply iterate through every exemplar in the region and find the cut value with the lowest variance, this is impractical. This implies that a significant number of iterations is required as the number of dimensions and exemplars increase. To improve the likelihood of finding the best cut value while keeping the number of iterations low, a new divide-and-zoom-in method is implemented to focus on promising ranges of cut values. The toy example in Figure 3.5 illustrates how this method works. This method can find a local minimum at each dimension that cleanly splits a region into two because it searches for cut value with the lowest variance by checking every single possible cut values within a range bounded by higher variance values. It keeps

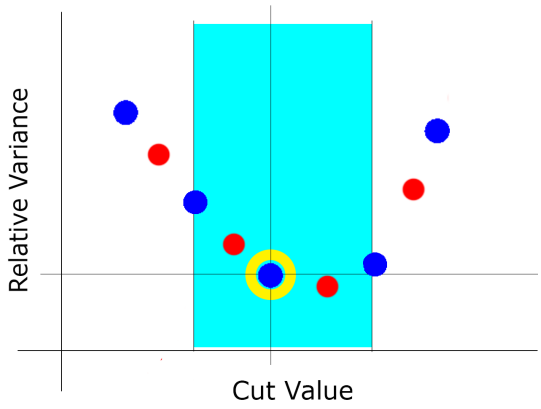
the number of relative variance computations low by focusing on promising regions where cut values with relatively low variances can be found.



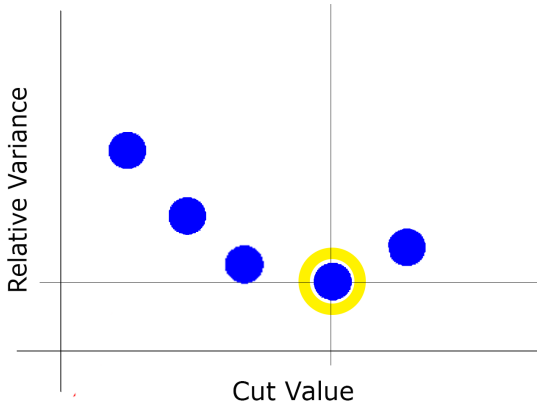
(a) At this stage, the relative variances have not been calculated yet. The red dots represent the actual relative variances if the region is cut in between each pair of exemplars.



(b) This is the first iteration of the toy example. First, compute the relative variances of N/k of the cut values (blue); then, select the point with the lowest relative variance (yellow ring); after that, identify the two adjacent points and zoom in to the range between those two points in the next iteration (light blue)



(c) This is the second iteration of the toy example. The computation is repeated in a smaller, zoomed-in range.



(d) Zooming in until the variance for every point within the range is computed.

Figure 3.5: An toy example explaining the "Divid-and-Zoom-In" region split values computation method by showing how relative variance changes with different cut values at each step.

Action Selection

Visually, it is difficult to distinguish between the periods when the CBLA is learning from the periods when the CBLA is simply executing random actions after all regions are learnt. In order to better demonstrate the learning process, we introduce some constraints that produce different actions given different levels of activation.

We considered two different types of implementations. In the first implementation, we introduced idle mode. The idle mode action is chosen as the action that requires the least power. This gives the visitors the sense that the sculpture is resting. During idle mode, the system selects the idle action a majority of the time. Otherwise, it will select an action based on its regular action selection policy. The CBLA engine enters idle mode when there is small knowledge gain potential. Conversely, it exits idle mode when the knowledge gain potential, or the change of it is higher than a threshold. Once it exits idle mode, it must stay in non-idle for at least a certain period of time before being allowed to enter idle mode again.

Alternatively, instead of having a hard threshold that moves the system from totally quiet to totally active, we used a sigmoid function shown in Equation (3.5) to define a range of activations from low activation to high activation. In this formulation, we capped the maximum output level m_{max} based on the knowledge gain potential x. This way, there is a visually lower level of activation while the knowledge gain potential is low.

$$m_{max} = \frac{1}{1 + e^{-k+a}} \tag{3.5a}$$

$$k = -\frac{1}{c}[\log(\frac{1}{d} - 1) - a]; \quad c > 0,\ 0 < d < 1 \tag{3.5b}$$

$$a = \log(\frac{1}{b} - 1); \quad 0 < b < 1 \tag{3.5c}$$

m_{max} is the maximum output level which can be a number between 0 to 1; x is the knowledge gain potential; b determines the minimum output level when knowledge gain potential is 0; c represents the x required for m_{max} to reach d.

In terms of how we compute the knowledge gain potential, we can either use average action value or relative action value. Average action value requires calibration and is not adaptive to fundamental changes in the system like when a sensor is covered or disconnected.

Relative action value is used to reduce the difficulty of tuning for different actuators as different actuators have different inherent level of action values range. This relative

action value is calculated by dividing the squared action value of this time step by average squared action values over the previous W time steps as shown in Equations (3.6) to (3.8). This means that even if there is a fundamental change to the system, it can still adapt over time.

$$Q_z = [q_{z-W}, q_{z-W-1}, ..., q_{z-1}] \tag{3.6}$$

Q_z is a set of previous W action values q at time step z; W is the window size and it is a constant.

$$\overline{q_z^2} = \frac{1}{W} \sum_{i=z-W}^{z-1} q_i^2 \tag{3.7}$$

$\overline{q_z^2}$ is the average squared action value at time step z.

$$\widehat{q_z^2} = \frac{q_t^2}{max(\overline{q_z^2}, \nu)} \tag{3.8}$$

$\widehat{q_z^2}$ is the relative action value; q_z^2 is the current action value; ν is the sensitivity constant.

One consequence of this formulation is that the denominator will become too small over time and a very small increase in action value can trigger large activation. This can be a desirable feature as it creates a system that periodically wakes up in hope to attract visitors when there is nothing to be learnt for long time. This sensitivity level during low learning period can be adjusted by changing the value of ν. On the other hand the window size W affects how quickly the system can adapt to a new ambient environment.

3.2 Multi-Node CBLA System

One option for controlling a larger system of sensors and actuators is to control them centrally through a single CBLA engine. However, if all output and input variables are grouped into one node, the sensorimotor state will become very large and it might take a very long time for the system to converge and react to changes. Therefore, subsets of related variables are grouped into nodes. Each node runs on its own CBLA engine in parallel. While our implementation uses a centralized high-level processor to compute these parallel CBLA engines, this architecture also allows for distribution of the software agents to local processors.

3.2.1 Sensorimotor Context of a CBLA Node

There are several options for grouping sensors and actuators into nodes. One approach is to group actuators and their associated proprioceptive sensors by function, because these relationships are the easiest to model. In addition, if multiple actuators work together to perform one function that directly affects one sensor, they should be grouped as a node because their relationship must be considered in order produce a prediction model for the sensor input. However, grouping simply by function cannot capture environmental changes and occupants' interaction effectively.

Another approach is to group sensor input and actuator output variables by spatial proximity. Since environmental changes and occupant interaction are likely to affect components that are close together physically, this will allow the system to capture those dynamics more effectively.

However, all sensor input and actuator output variables associated with a particular node are updated once per loop. This means that the loop speed is limited by the slowest actuator or sensor within the node. For instance, an SMA-actuated Fin node has heating and cooling cycle time of about 12 seconds, while an LED can be turned on and off many degrees of magnitude faster. This is very inefficient and limiting for the fast components. Therefore, components running at different speeds should be grouped into different nodes. This allows each node to run at its own speed, independently from the other nodes.

In our implementation, each node is constructed of components that are related functionally and proximally. For instance, LED output and ambient light sensor are in one node; Fin motion and accelerometer are in one node. Different nodes run at different frequencies and capture the dynamics of the system at different time scales. We limited the number of outputs per node to one. This structure keeps the system's dimensionality low and flexible. This also allows us to optimize the CBLA loop rate for each type of actuator as the bandwidth is mainly constrained by the bandwidth imposed by the actuators. Also, no more than one node can control one physical actuator since this will result in conflicts.

3.2.2 Inter-Node Connections

Nevertheless, if the nodes perform learning independently, the CBLA does not actually capture the dynamics of the entire system. To integrate the nodes into one coherent entity, we have used shared sensors and virtual inputs.

By sharing input variables that share the same sensor, system changes effected by one node can be detected by another node indirectly. For instance, two adjacent nodes might

share a single IR Proximity sensor. This way, change in the environment can be seen by both nodes. However, when two Nodes share one input variable, if the action of the actuator does not affect the shared sensor, information about that event is not transferred to the other Node.

Another method is to treat output signal from a node as input to other nodes. We called this virtual input. This way environmental changes happening to another node can be detected by other nodes indirectly in the time steps afterwards. The idea is that external disturbances, like a user standing in front of the sensor of a node, may be represented as an output signal originating from that node, which may be measured by another node. This gives the other node information about the fact that there is someone nearby despite being somewhat delayed. Figure 3.6 illustrates how Node 2 can detect changes in the environment that are not covered by its own sensors through Node 1.

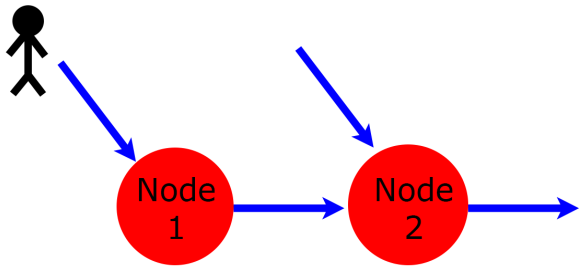


Figure 3.6: An example of illustrating how Node 2 can indirectly detect the presence of a user through Node 1. The presence of the user may affect the output of Node 1. Since Node 2 takes the output of Node 1 as input, Node 2 would in effect respond to the presence of the user indirectly despite not having a sensor that can detect the user in its sensorimotor context.

3.2.3 Summary of CBLA

This chapter details the implementation of the CBLA Engine, which is an reinforcement learning algorithm that selects actions that lead to the maximum potential knowledge gain. Each Engine represents a subset of the system and constructs a model that predicts the consequences of its actions given its current states. A system is then formed by connecting a network of CBLA Engines through shared sensors and virtual inputs. The distributed approach enables the system to respond to the occupants and adapt to localized changes

in the surroundings quickly. Building on this CBLA system, a sculptural system that generates its own life-like, interactive behaviours may emerge.

Chapter 4

Interactive Control System [1]

The CBLA requires a large increase in sensing and control capability which the previous versions of the Hylozoic Series embedded electronics [3] are unable to provide. A new Hylozoic Series 3 interactive control system was developed with re-designed hardware, to enable the control and sampling of a larger number of actuators and sensors at a higher frequency.

To enable the sculpture to understand its own mechanisms, in addition to its surrounding environment and the occupants, there must be at least one proprioceptive sensor associated with each actuator. Note that while the nominal purpose of such a sensor is to directly sense the results of activation, they can also be used to detect changes in the environment.

This chapter discuss the design of the hardware and software mechanism that enable the learning algorithm behaviours, and some of the specific actuator and sensors that we use.

4.1 System Architecture

In this implementation, we opted for an architecture where a centralized computer controls a number of smaller localized microcontrollers that interface with the sensors and actuators. By running the high-level algorithm on a central computer, complex algorithms can quickly

[1] Part of this chapter is adapted from papers published with IROS 2015 [1] and Next Generation Building [2]

be developed without the constraints of a embedded platform. In addition, it provides an abstraction layer that enables the designer to build virtual sub-systems using any of the sensors and actuators. These two features allow us to quickly test out different configurations. On the other hand, by distributing the low-level logics on localized microcontrollers, the system can be more modular and the number of wires can be reduced. In addition, the real-time nature of the microcontrollers allows protection of sensitive actuators in case of central computer failure.

Figure 4.1 shows the high-level system architecture of the Hylozoic Series 3 Interactive Control System. Each component runs asynchronously. At the bottom, there are the Teensy Devices. They represent the physical components in the sculptural system. Each Teensy Device consists of a Teensy Development Board[2] that communicates with a computer which hosts the abstract components via Universal Serial Bus (USB). Each Teensy controls and samples from a number of actuators and sensors. More details about the physical hardware are presented in Section 4.2.

For each Teensy Device, a Teensy Interface thread is created with the corresponding communication protocol and system parameters. Each type of Teensy Device has its own unique set of parameters that dictates or reflects the behaviours of its actuators or sensors respectively. Examples of some of the parameters are the brightness of an LED, and the readings from an accelerometer. The InteractiveCmd module was created to provide a communication layer for all the physical components. Nodes can be created to control or sample any of the actuators and sensors through the InteractiveCmd module. The InteractiveCmd module handles the commands sent from the high-level algorithm and forwards them to the appropriate Teensy Interface in an efficient manner. A Messenger was created to streamline the message delivery system by periodically pushing commands received from the abstract Nodes in a more compressed form. This reduces the number of commands being sent to and received from InteractiveCmd and, thus, enables the system to operate more efficiently.

The Input Nodes and Output Nodes are the abstractions for sensors and actuators. Their sole purpose is to sample or update a particular sensor or actuator continuously. The second level of abstraction are the Device Nodes. Each Device Node is generally made up of one or more Input or Output Nodes, though it can also be completely virtual. It provides additional functionalities such as providing progressive dimming effect to the LEDs, or calculating the running average of a variable. The third level of abstraction are the CBLA Nodes. They are made up of components of the Device Nodes. This is also where the CBLA Engine and Prescripted Engine reside. The internal working of the CBLA

[2] Teensy Development Board: www.pjrc.com/teensy/teensy31.html

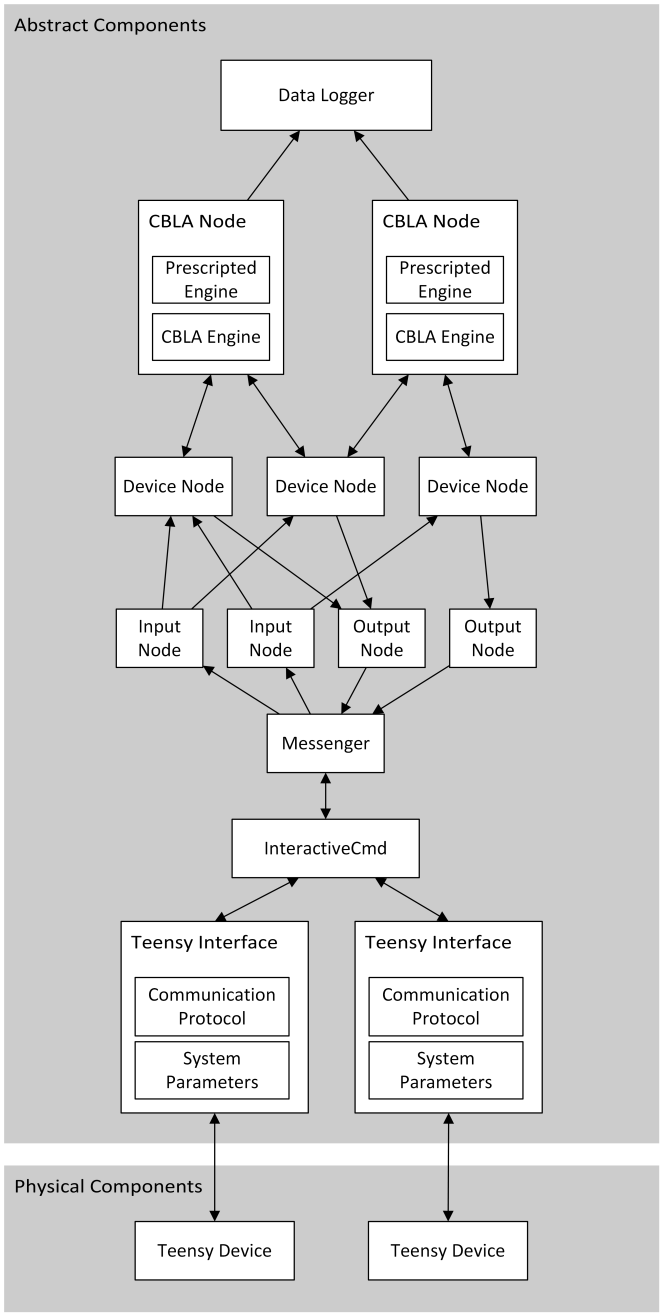


Figure 4.1: The system is comprised of abstract and physical components. Each component runs asynchronously in its own thread. Each arrow represents the data flow from one component to another.

Engine is discussed in Chapter 3. A Prescripted Engine is an alternate set of behaviours that follow user designed scripts, and do not involve any learning algorithm. A CBLA Node can be switched to either Engine while it is running.

The Data Logger is a special kind of abstract component that periodically samples the abstract variables of each abstract node, packages the data in the memory, and saves data in the hard drive in an efficient manner. The data collected are stored in a persistent key-value database for further analysis. In addition, the entire state of the CBLA Engine is also saved. This allows the CBLA system to recover from a previous state at a later time.

4.2 Hardware

The hardware was custom designed in collaboration with Mohammadreza Memarian. The goal is to develop a system that is modular, flexible, and expandable. Analogue sensors and actuators that are traditionally used in previous Hylozoic Series as well as off-the-shelf sensors that interface via I2C, SPI, or UART are supported. Most importantly, the design enables high-speed two-way communication with a computer over USB. Sensor readings and control signals can be delivered to and from a computer at around 100Hz. This capability allows us to run algorithms that are more computationally intensive; simplify the implementation of multi-threaded and multi-process software; and collect and display a large amount of data at runtime on a standard computer.

4.2.1 Actuators and Sensors

The sensors and actuators used in the experiments described in Chapter 5 and their interface types are tabulated in Table 4.1 and Table 4.2 respectively.

Table 4.1: List of sensors that were used in the experiments and their interface types

Sensor	Interface Type
IR proximity sensor[3]	Analogue
Accelerometer[4]	I2C
Ambient light sensor[5]	Analogue

Table 4.2: List of actuators that were used in the experiments and their interface types

Actuator	Interface Type
Shape memory alloy (SMA) wire[6]	PWM
Vibration motor	PWM
LED	PWM
High-power LED[7]	PWM

The aforementioned actuators and sensors interface with the Control Module through some custom drivers. We call these drivers Device Modules. Different actuators or sensors require different Device Modules due to their different power requirements. They are plugged into the Device Ports on the Control Modules (discussed in Section 4.2.2). These Device Modules provide the power switching and voltage regulation needed to drive the actuators and sensors.

Combinations of the actuators and sensors listed in Tables 4.1 and 4.2 are used to form three functional units: 1) Fin Unit, which is composed of two SMA wires, one infrared (IR) proximity sensor, and a three-axis accelerometer; 2) Reflex Unit, which has a pair of LEDs (wired in together), a vibration motor, and one infrared (IR) proximity sensors; and 3) Light unit, which comprises a high-power LED, and an ambient light sensor. Their spatial configurations are shown in Figure 4.2

A Fin Unit moves a 2-DOF Fin-like mechanism made of flexible plastic rods by controlling its two SMA wires. A Reflex Unit actuates a vibration motor and mobilizes the Frond-like mechanism attached to it. The pair of LEDs shine on to the Frond for additional effect. A Light Unit contains a very bright LED and it is hung high above the sculpture. The IR proximity sensors detect the position of the occupants or objects within their ranges. The accelerometers detect contacts with the occupants and the deformations of the Fins. The ambient light sensors detect the light intensities of the ambient environment and their associated LEDs.

[3] Sharp GP2Y0A21YK Infrared Proximity Sensor: www.sharpsma.com/webfm_send/1208
[4] ADXL345 3-Axis Digital Accelerometer: www.analog.com/media/en/technical-documentation/data-sheets/ADXL345.PDF
[5] SparkFun Ambient Light Sensor Breakout (TEMT6000): www.sparkfun.com/products/8688
[6] Dynalloy Flexinol Actuator Wire: www.dynalloy.com/tech_data_wire.php
[7] Indus Star High-Power LED Light Module: www.luxdrive.com/content/A007_A008_Data_Sheet_V1.2.pdf

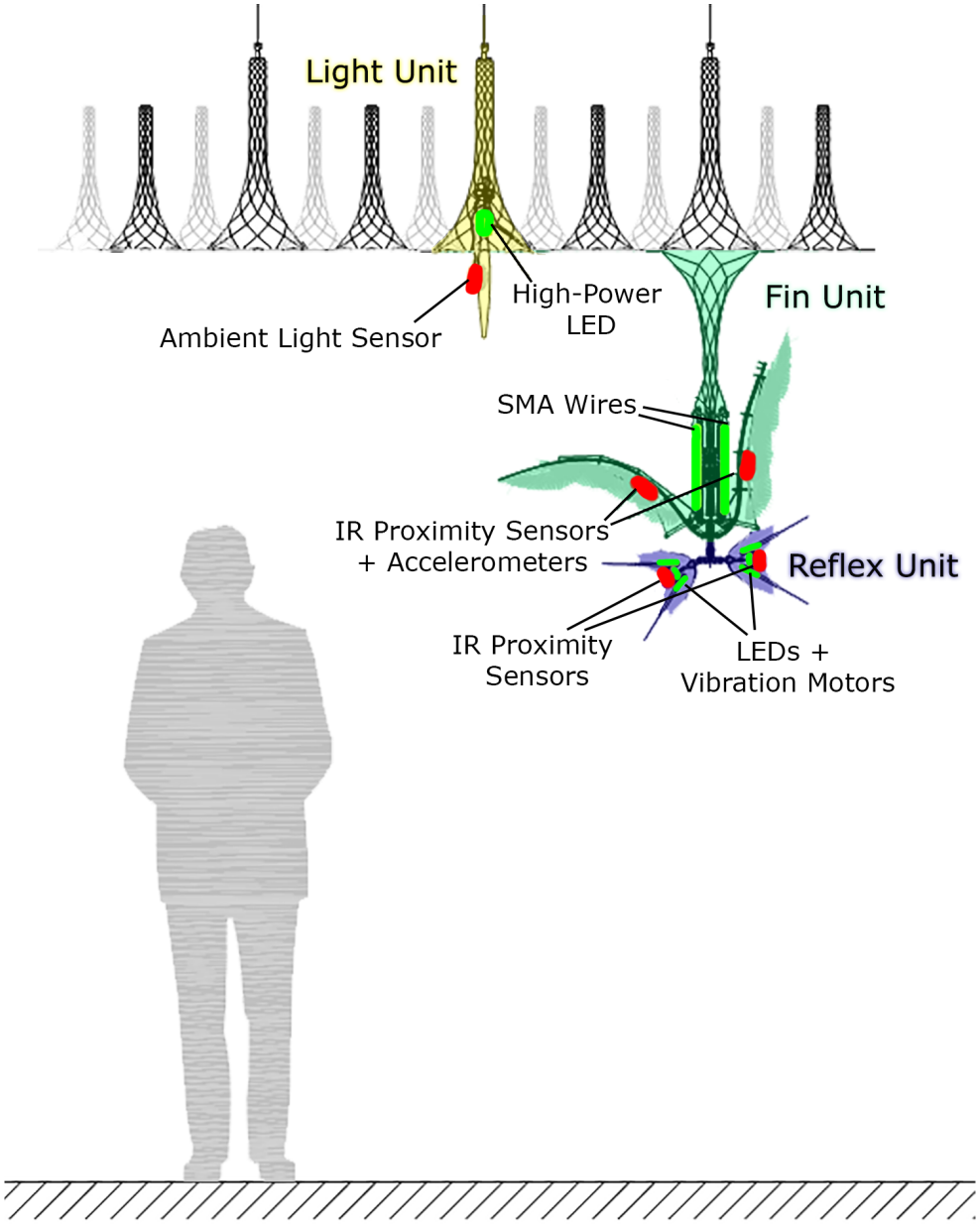


Figure 4.2: This is a typical spatial configuration of the function units (figure adapted from an image provided by Philip Beesley Architect Inc.). Bright green denotes actuators and bright red denotes sensors. Light Unit is shaded yellow; Fin Unit is shaded green; and Reflex Unit is shaded blue.

4.2.2 Control Modules

At the heart of a Control Module, shown in Figure 4.3, there is a Teensy 3 USB-based development board. It controls each device through the Device Port and communicates with the computer through USB as a human interface device (HID). Each Device Port has four output pins capable of pulse-width modulation (PWM), two input pins connecting to the Teensy's on-board analogue-to-digital converter (ADC), and two lines for digital serial communication over Inter-Integrated Circuit bus (I2C). It is designed to provide convenient and fast interface to commonly used analogue actuators and sensors, and a digital bus for added flexibility. The on-board SPI and UART port are reserved for future expansion.

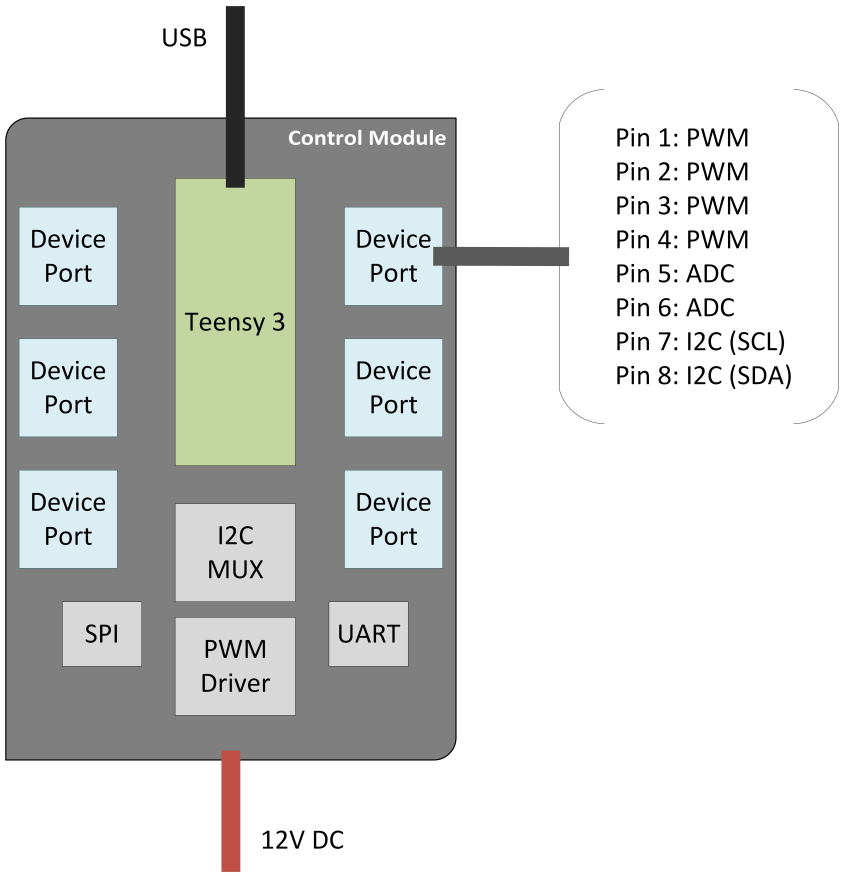


Figure 4.3: A Control Module consists of a Teensy 3, 6 Device Ports, a SPI port, and a UART port. Each Device port has 8 wires and they carry the signals that are commonly used in our system.

In order to increase the number of PWM pins beyond what is provided natively on the Teensy, a I2C bus controlled PWM controller[8] was used. This does not affect the I2C buses on the Device Ports since it is using the other one of Teensy's two I2C buses. The I2C bus for the Device Ports is further multiplexed into six. This makes each Device Port more independent, and devices may have the same addresses as long as they are on different Device Ports. By having virtually six independent I2C buses, it simplifies the configuration of the Device Modules as they can all be configured the same way.

4.3 Control Software

The control software of the sculpture consists of a low-level layer and high-level layer. The two layers are connected via USB. The low-level layer consists of the firmware that interfaces with the peripherals that connect with the actuators and sensors. The high-level layer consists of the tools that facilitate communication between the two layers and the application that dictates the system behaviour. The abstraction provided by the high-level layer allows flexibility in defining the nodes and their relationships to each other.

4.3.1 Communication and Interface

A low-level layer of firmware written in C++ runs on the Teensy 3 USB-based development boards which interface with the peripherals that connect with the actuators and sensors. High-level software written in Python 3.4 is referred to as applications, and runs on a central computer. The use of the central computer as a development platform provides flexibility for development free from the limited processing power and specialized functions inherent to the Teensy microcontroller hardware. Moreover, Python 3.4 is cross-platform and supports multi-threading, permitting operation within many operating systems and allowing multiple sets of software instructions to be executed in parallel. Code that is necessary for communicating with the low-level layer is packed into a Python Package named *interactive-system*. Developers can then develop applications that control and retrieve information from the sculptural system firmware using the software utilities provided by the Python Package. Each application can run on its own thread. While care should be exercised to avoid conflicts among threads, this should permit multiple applications to execute simultaneously.

[8] PCA9685 16-channel, 12-bit PWM Fm+ I^{2}C-bus LED controller: www.nxp.com/documents/data_sheet/PCA9685.pdf

CBLA executes as an application that communicates with the low-level layer using the *interactive-system* Python package. Other applications such as an occupancy map that uses the sensors on the sculpture to interpret the locations of the occupants can run simultaneously, taking advantage of the multi-threading properties of the high-level platform.

Figure 4.4 illustrates how an application communicates with the Teensy devices. At the high-level layer, the Teensy Interface module in the *interactive-system* package is used to create a thread for each Teensy device. The thread looks for changes in those parameters and performs synchronization. Each Teensy device on the low-level layer is represented by a Teensy Interface thread on the high-level layer. Teensy devices are considered as slave devices in this communication mechanism. Only the Teensy Interface, the Master, can initiate a read or write request. An InteractiveCmd thread can modify a Teensy's output parameters and retrieve its input parameters through its Teensy Interface.

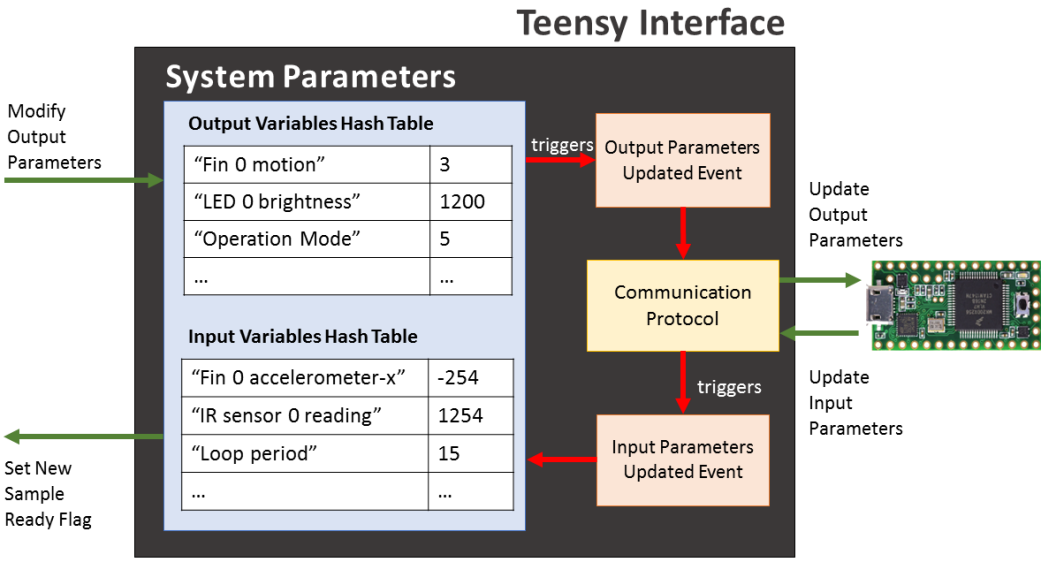


Figure 4.4: Teensy Interface is the connection between the InteractiveCmd and its associated Teensy device. InteractiveCmd modifies output parameters on its corresponding Teensy Interface's output hash table. This triggers an event that pushes the changes to the Teensy device. The Teensy device would then respond by triggering an event that updates the input hash table with newly sampled input values and notifies the InteractiveCmd.

4.3.2 Node Abstraction

Between the Nodes and the Teensy Interface, there is the InteractiveCmd. Its job is to forward messages to the correct Teensy Interface and hide the physical implementation of the low-level layer devices from the Nodes. In addition, since the InteractiveCmd module enables the control and sampling of any actuators and sensors in the system, a Node can be constructed unconstrained by spatial or hardware specificities. Each Node, physical or virtual, is represented by a set of input and output variables which can be accessed by any other nodes in the system, and each runs continuously in its own thread. Input variables are simply variables in the memory that Nodes have read access to. Similarly, output variables are variables that Nodes have write access to. Multiple Nodes can share one input variable while only one Node can be associated with one output variable. Figure 4.5 illustrates the relationships between different Nodes.

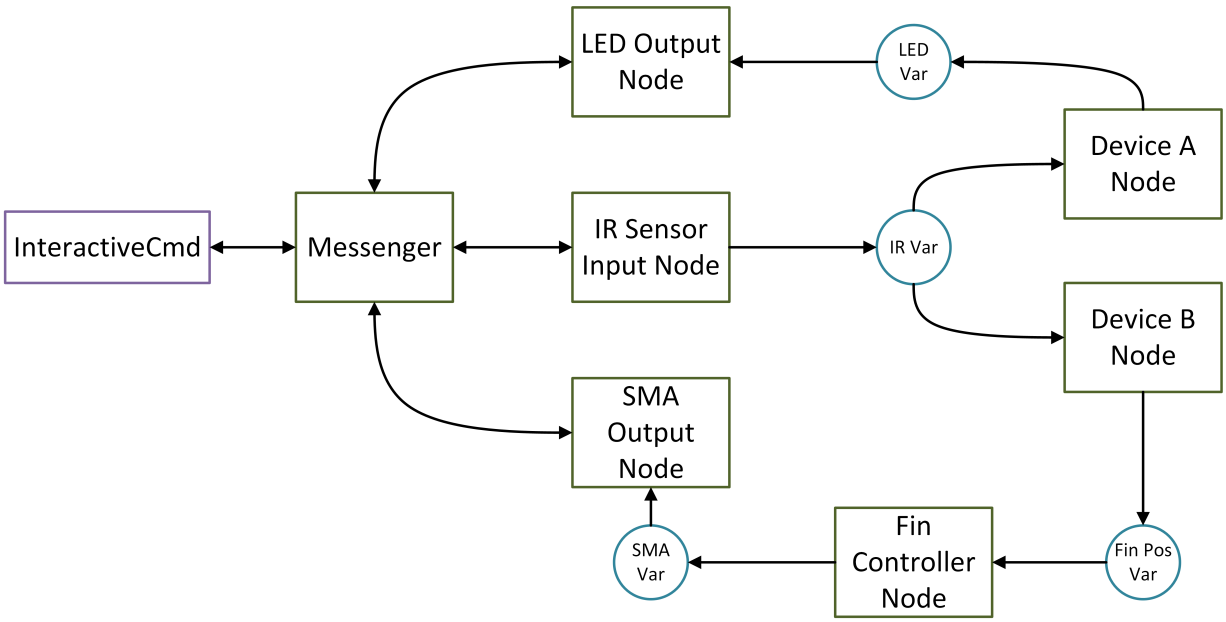


Figure 4.5: Illustration of how abstract nodes work. Each rectangle represents a thread. The purple one represents InteractiveCmd module and each green one represents a Node. The blue circle represents a variable in the memory. The arrow represents the direction of access. Multiple Nodes can share read access to a variable while only one Node can have write access to one variable.

At the lowest level, Input Nodes continuously update their associated variables and

Output Nodes continuously send output requests to InteractiveCmd through the Messenger. Different types of Input and Output nodes are configured to run at a loop period compatible with the physical components that they represent. This mechanism makes implementation of the higher level Nodes much easier by eliminating the need of communicating with the InteractiveCmd by means of sending individual messages. Instead, each of the input and output variables can be accessed from the memory at any time. These variables are used as building blocks for higher level Nodes. In addition, intermediate level Nodes can embed extra functionalities. For instance, a LED Driver ramps up or dims an LED to the desired brightness level. A higher-level Node controlling that LED using the LED Driver can then operate at a lower update period and process more complex logic. This Node Abstraction system makes developing CBLA Nodes much simpler by eliminating the need for managing logic requiring different frequencies of control under one thread.

The addition of the Messenger node between the InteractiveCmd, and Input and Output Nodes streamlines the communication by reducing the number of messages. Over USB, each packet can contain up to 64 bytes. If each Node communicates with the InteractiveCmd directly, there will be many messages that might only require one or two bytes. A large portion of the communication bandwidth will be wasted and the update rate of the Nodes will be significantly throttled. Since many messages are likely to be delivered to the same Teensy, those messages can be combined and delivered as a single packet. The job of the Messenger Node is to collect all the messages, combine them appropriately, and deliver them to the InteractiveCmd periodically. Although this means that each message must wait for the next delivery cycle to be sent out, this mechanism allows the system to handle a much higher throughput. To avoid commands or requests being missed, the rate of each Input or Output node is set to be at least three times the Messenger's update period.

4.4 Data Logging

For secondary analysis, the values of all input and output variables of every Node, as well as the internal variables within each CBLA Engine must be collected and stored onto the hard drive. In addition, the state of the CBLA Learner including all the exemplars, the prediction models, and the definitions of the regions must be stored such that it can be recovered at a later time.

The CBLA system contains a large number of asynchronous threads that run at their own speeds. As a result, a large amount and variety of data are generated at high frequencies and at different times. These data must be handled in a way that does not slow

down the system. In addition, in case when the program fails to terminate safely, the majority of the data should still be recoverable. Also, the data must be saved to disk and be discarded from the memory continuously as the CBLA system is expected to operate for a long period of time.

4.4.1 Key-Value Database

The data generated by the Nodes are in many different types, such as integer, floating point, string, list, tuple, and other custom object types. In addition, objects such as the CBLA Learner are continuously expanding and its hierarchy gets deeper over time. Moreover, each type of data packet gets generated at different non-constant time cycles. This makes simply saving them in a table or a relational database impractical. In our implementation, a simple key-value type NoSQL database based on Python's shelve[9] module was used. In shelve, the data are stored as serialized Python objects using the pickle[10] module. This type of database gives the flexibility of storing an assortment of data types without the need of predefining them. However, since the data are not stored as plain text, a special script is required to extract the data in the desired formats for offline analysis.

4.4.2 Database Structure

Each time the CBLA System is run, a new shelve database is created for the session. This is to ensure that data from previous sessions would not get corrupted accidentally. In addition, in order to restart from a previous session, one only has to remove the database files of the succeeding sessions. If there are previous sessions, the CBLA system will access an index file to locate the database file of the most recent session and retrieve information regarding the previous state of the system. Information about the start time, end time, and the configurations of the system can also be found in the index file. Figure 4.6 illustrates the structure of the database created by the Data Logger.

For each Node, there are two main types of data: Packet Type and Info Type. Each Packet Type data has a timestamp which indicates when the data block is generated. These data blocks are generated continuously and they must all be stored. It contains information about the sensor readings, actuator outputs, and the internal parameters of the CBLA Engine such as its current mean error, number of regions, and relative action values. Packet Type data are mainly used for secondary analysis purposes. On the other

[9] Python shelve documentation: https://docs.python.org/3.4/library/shelve.html
[10] Python pickle documentation: https://docs.python.org/3.4/library/pickle.html

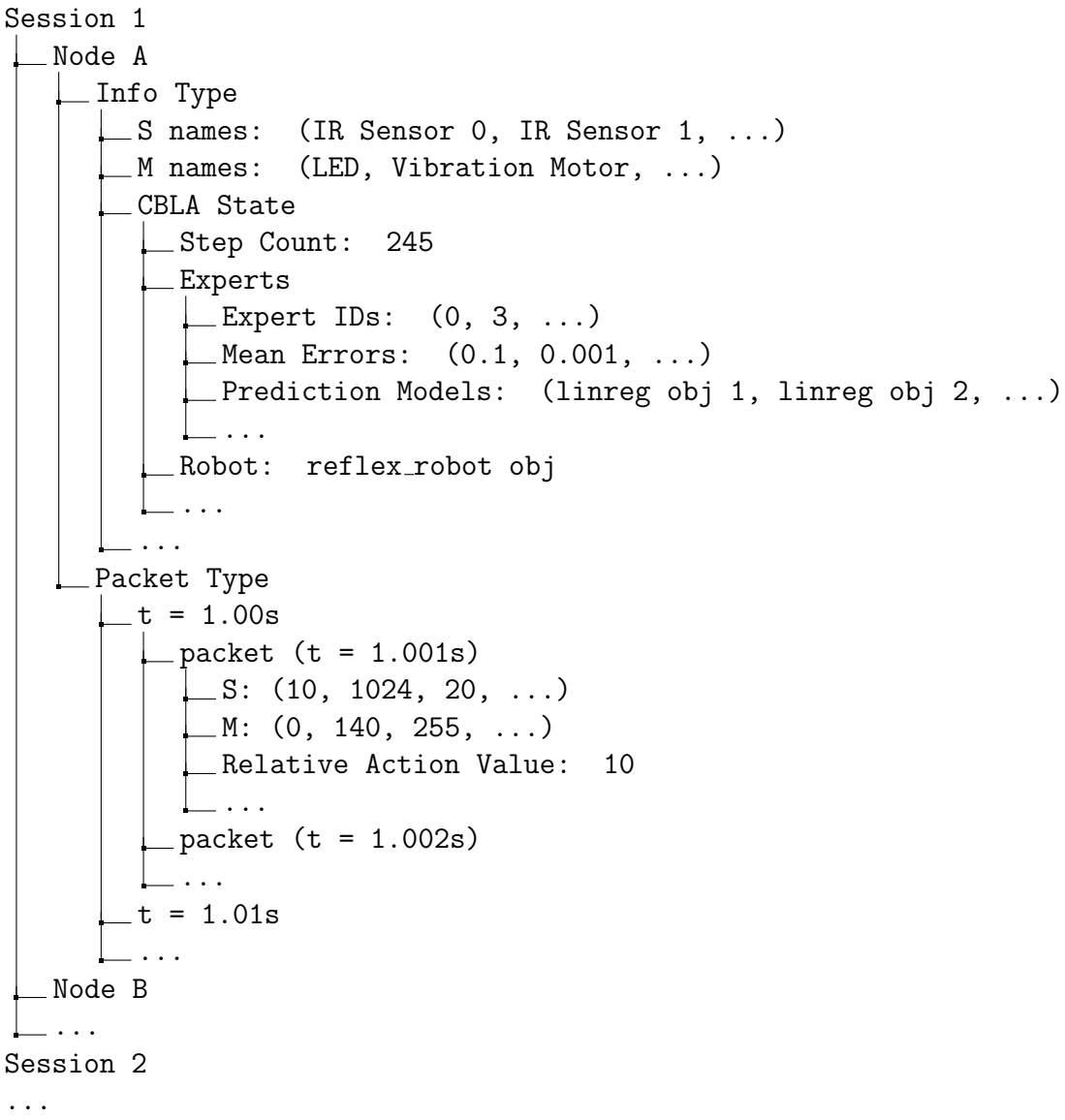


Figure 4.6: Example showing the structure of the database created by the Data Logger .

hand, Info Type data describes the system and does not have a timestamp. When new versions of Info Type data arrive, the old ones can be overwritten. Information like the names and order of the sensor and actuator variables, which do not change over the runtime of the system, are Info Type data. In addition, the state of the CBLA Engine, which is needed for recovering from a later time, is also an Info Type data. Over the long term, while it is desirable for the state to be saved frequently, the size of the data that describes it would become too large to allow multiple copies of it to be stored. By saving it as an Info Type, the old version can simply be overwritten by a newer version.

4.4.3 Data Logging Process

Due to the large amount of data that are being generated at high frequencies, saving each block of data directly would require too much time and introduce significant lag in the system. Therefore, a multi-stage process as illustrated in Figure 4.7 is used to improve data logging efficiency.

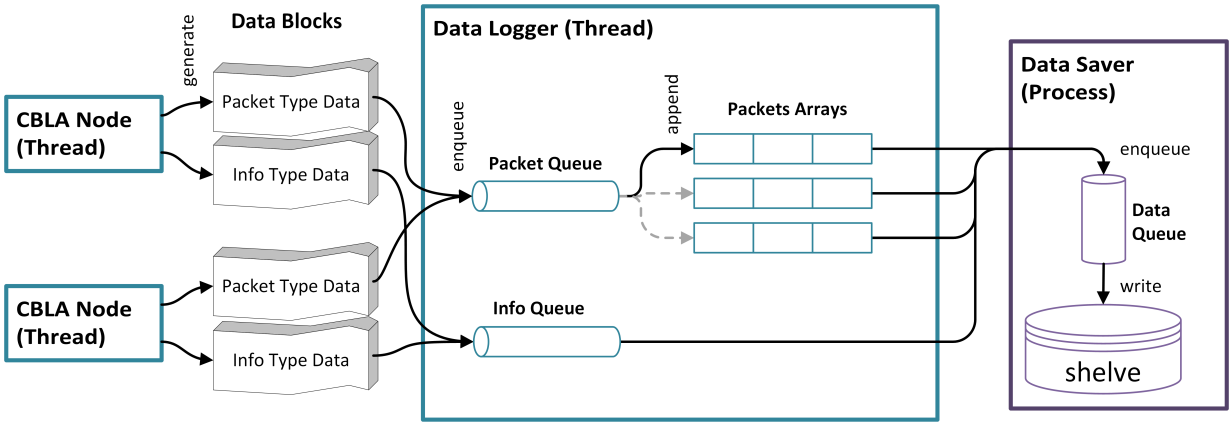


Figure 4.7: Flowchart of the data logging process from data generation to writing to the disk.

Packet Type data blocks are being generated at a rate of approximately one block every 20 milliseconds per Node. This is relatively high given the hundreds of Nodes that a typical CBLA system has. On the other hand, Info Type data are being generated at a much slower rate at roughly one data block every two minutes per Node. Both types of data blocks would first get enqueued onto the Data Logger before being transferred to the Data Saver which is the module responsible for writing the data into the shelve database. This step is necessary to avoid slowing down the CBLA Nodes because Data Saver is a

separate process, and transferring data to another Process takes significantly longer time than to another thread.

For the higher frequency Packet Type data blocks, instead of directly being sent to the Data Saver, they are first packed in Packet Arrays. This process decreases the total transfer time to the Data Saver by drastically lowering the number of enqueue calls which have non-trivial overhead. There is one Packet Array for each Node and they get enqueued to the Data Saver periodically.

The Data Saver is implemented as a process in order to avoid the GIL limitation imposed by Python [25]. This enables the system to make use of the parallel computing capability of a multi-core computer. Although transferring data to a process takes longer as it requires the copying of the actual data rather than just the pointers, it is still faster than writing the data onto the disk. Therefore, it is still more efficient to move the data and let a separate process load the data onto the database.

Nevertheless, depending on the number of Nodes in the system, there are situations when data are indeed being generated at a rate faster than it can be stored. Eventually, over the long term, the memory of the host computer will be full and the program will crash. In addition, a long wait time between data generation and data storage means that if the program crashes unexpectedly, a large amount of data would be lost. To avoid crashing and the loss of data, once the length of the queue has reached a certain threshold, a clean-up procedure is activated. It momentarily pauses all other threads, and allocates all the processing power for the purposes of data storage. Practically, this process would only takes around 50 to 70 milliseconds. Since it happens only once every few minutes, it is generally not noticeable by human viewer. In fact, the reason why it only takes so little time is because, by pausing all other threads, it eliminates the overhead of thread switching. This process ensures that the CBLA system can operate as long as there is sufficient storage space on the hard drive of the host computer.

4.5 Multi-Cluster Test Bed

An experimental test bed was built to investigate how users interact with the CBLA system. This is a small scale four-cluster test bed that resembles a typical interactive sculpture produced by Philip Beesley Architect Inc. (PBAI) and it was used in the experiments described in Sections 5.3 and 5.4. A photograph of the complete test bed is shown in Figure 4.8

Figure 4.8: Photograph of the multi-cluster test bed when actuated. The names of the Clusters from left to right are: Cluster 1 ($C1$), Cluster 2 ($C2$), Cluster 3 ($C3$), and Cluster 4 ($C4$).

4.5.1 Electronic Components

A Light Unit is made up of one high-power LED and one ambient light sensor. The high-power LED is mounted on top of a flask containing coloured liquid. The ambient light sensor is mounted on the side of the flask under the LED. This allows the ambient light sensor to measure the intensity of the light emitted by the LED.

A Fin Unit is made up of two SMA wires, a pair of LEDs, a vibration motor, a 3-axis accelerometer, and two IR proximity sensors. The two SMA wires pull on two levers that move a Fin, which is a mechanism made of soft plastic rods that curls up. An IR proximity sensor and an accelerometer are mounted at around midway between the tip to the root of the Fin. At the bottom of the Fin, a vibration motor, a pair of LEDs, and an IR proximity sensor are mounted in the middle of two frond-like objects.

4.5.2 Device Nodes

Device Nodes further abstract the Output and Input Nodes to enable higher level functionality. This frees the CBLA Nodes from managing the constraints imposed by the physical attributes of the actuators and sensors.

SMA Controller Node

In the experiment described in Section 5.2, the SMA wires were only operated in fully-off or fully-on mode. This means that a Fin with two SMA wires can only have four possible states. In addition, each actuation must be a cycle since the SMA wires cannot be fully actuated at 5V for more than 2 seconds. Empirically, we determined that the cooling period takes around 10 seconds. This means that the loop period for a CBLA Node cannot be lower than 12 seconds since it does not have the freedom to actuate the SMA wires again during the cooling period. These restrictions are problematic during interaction with the users since the learning period and response latency would likely take longer than what a typical visitor would spend in front of a section of a sculpture. In addition, only a very coarse model can be made with only four possible actions that the Fin can choose from. This means that the Fin Node would likely be very unresponsive since the variance in resultant state for each kind of action is likely to be very high.

Thus, for this experiment, a position controller Node is needed to enable the SMA wire to hold its contraction while keeping the SMA wire at a safe temperature. Since the length of an SMA wire is related to the temperature, and current is passed through the SMA wire to generate heat, to maintain a position, the controller needs to adjust the output voltage to a level that can maintain the desired temperature. This can reduce the loop period as the SMA wires no longer need to cool down after actuating. In addition, the number of actions is no longer limited to four as the Node can select any value between fully-off and fully-on. However, it is difficult to attach a temperature sensor on the SMA wire. Thus, instead of using a feedback controller, only an open-loop controller with a model that estimates the temperature of an SMA wire is used.

This controller essentially produces a control signal that allows the SMA wire to quickly reach its desired temperature by setting the output voltage very high. It then gradually lowers the voltage as the SMA wire reaches its desired position according to an internal model. Development of this model starts with the intuition that the temperature of the wire increases when the rate of heating is greater than the rate of cooling. The rate of heating is related to the voltage across the wire. On the other hand, the rate of cooling is

related to the temperature of the wire since it is mainly driven by the wire's own natural convection.

We first calculate the heat transfer rate as a result of current passing through the SMA wire. According to Joule's Law [26],

$$q_1 = i^2 \cdot r \tag{4.1}$$

where q_1 is heat transfer rate in Watts; i is the current in Amperes; r is the resistance of the wire in Ohms.

According to Ohm's Law,

$$i = \frac{v}{r} \tag{4.2}$$

where i is the current; v is the voltage in Volts; and r is the resistance.

Substituting (4.2) into (4.4), we get

$$q_1 = \frac{v^2}{r} \tag{4.3}$$

In our case, although r is not actually a constant since the resistance of the SMA wire decreases as it shortens [27], the effect is sufficiently small that it can be treated as a constant. Therefore, we get,

$$q_1 = k_{heating} \cdot v^2 \tag{4.4}$$

where q_1 is the heat transfer rate in Watts.

Since our control signal, x, is proportional to the voltage, it can simply be absorbed into the $k_{heating}$ constant as

$$q_1 = k_{heating} \cdot x^2 \tag{4.5}$$

Then, we calculate the heat loss rate due to natural convection of the SMA wire. According to Newton's Law of Cooling [28],

$$q_2 = h_c \cdot A \cdot dT \tag{4.6}$$

where q_2 is the heat transfer rate in Watts; h_c is the convective heat transfer coefficient; A is the area of the heat transfer surface in m^2; and dT is the temperature difference between the air and the surface in Kelvin.

If we approximate h_c, A, and the air temperature T_{air} as constants, we get

$$q_2 = k_{cooling} \cdot (T - T_{air}) \tag{4.7a}$$
$$= k_{cooling} \cdot T - k_{cooling} \cdot T_{air} \tag{4.7b}$$
$$= k_{cooling} \cdot T + k_{air} \tag{4.7c}$$

where q_2 is the heat transfer rate in Watts; $k_{cooling}$ and k_{air} are constants; and T is the temperature of the SMA wire in Kelvin.

Combining (4.5) and (4.7c), we get the total heat transfer rate as

$$q = q_1 - q_2 \tag{4.8a}$$
$$= k_{heating} \cdot x^2 - k_{cooling} \cdot T + k_{air} \tag{4.8b}$$

We can then calculate the kinetic energy generated during time interval Δt by multiplying (4.8b) by Δt.

$$KE = (k_{heating} \cdot x^2 - k_{cooling} \cdot T + k_{air}) \cdot \Delta t \tag{4.9}$$

Since temperature is directly proportional to kinetic energy, the proportionality constant can be absorbed into $k_{heating}$, $k_{cooling}$, and k_{air} as well.

$$\Delta T = (k_{heating} \cdot x^2 - k_{cooling} \cdot T + k_{air}) \cdot \Delta t \tag{4.10}$$

At each time step, Δt, the temperature is incremented by ΔT.

$$T_{t+1} = T_t + \Delta T \tag{4.11a}$$
$$= T_t + k_{heating} \cdot x^2 - k_{cooling} \cdot T_t + k_{air} \tag{4.11b}$$

In this formulation, the actual unit of the temperature is not important. Instead, we define 0 as the temperature when the SMA wire is the longest and 1 when the SMA wire is the shortest. The steady-state temperature of the SMA wire should be 0 when the input, x, is 0. If we substitute 0 into T_{t+1}, T_t, and x into (4.11b), we get $k_{air} = 0$.

$$T_{t+1} = T_t + k_{heating} \cdot x^2 - k_{cooling} \cdot T_t \tag{4.12}$$

From the SMA wire's technical specifications[27], we identify the maximum current at which the SMA wire can operate continuously without heat damage. Using that value, we

determine the maximum continuous output level, x_c. This means that when the desired temperature, T, is equal to 1, at steady-state, x should be at x_c. This means,

$$0 = k_{heating} \cdot x_c^2 - k_{cooling} \tag{4.13a}$$

$$k_{cooling} = k_{heating} \cdot x_c^2 \tag{4.13b}$$

The SMA wires would not be damaged due to over-heating as long as the relationship in (4.13b) holds. In fact, we can set $k_{heating}$ to 1 arbitrarily to simplify (4.12) further. At the end, we get

$$T_{t+1} = T_t + x^2 - x_c^2 \cdot T_t \tag{4.14}$$

as the temperature model for the SMA Controller Node.

We then apply and tune a PI controller on this temperature model to track a desired temperature as shown in Figure 4.9. A Node can specify a desired temperature, $T_{desired}$, between 0 and 1 and the SMA Controller tracks this temperature within its internal temperature model. The control signal generated by this PI Controller is applied to the actual SMA wire in parallel.

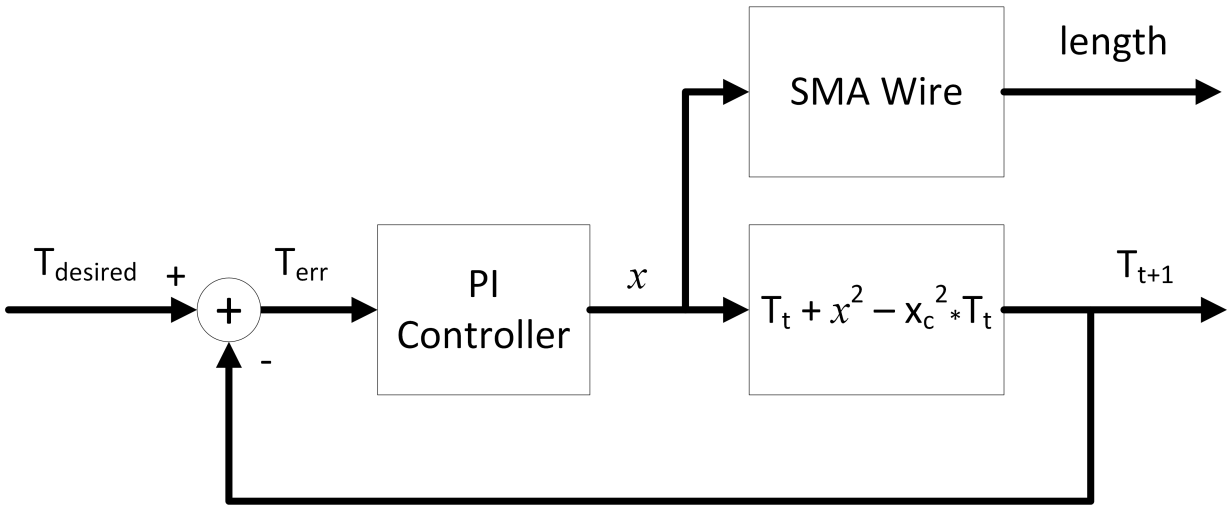


Figure 4.9: Block diagram of the SMA Controller.

Due to the lack of feedback control and the simplifications and approximations made when developing the model, this controller is unlikely to be accurate. However, the main purpose of this SMA Controller is to enable the CBLA Node to hold the Fin at a particular position. As a result, this allows the CBLA Engine to run at a higher rate and expands the number of possible actions.

LED Driver Node

If a CBLA Node controls an LED directly, any change in brightness would be set instantly. As a result, the LED may appear to be flicking or flashing erratically to the viewers as it rapidly jumps between different brightness levels. To improve the aesthetic, an LED Driver that brightens and dims the LED gradually is introduced. However, the relationship between brightness level of a typical LED and the input voltage is non-linear [29]. At low voltage levels, the brightness increases rapidly as input voltage increases. At higher voltage levels, a larger increase in voltage is needed to increase the brightness at the same rate. In order for the change in brightness to appear more linear, (4.15) is used to update the output level of an LED.

At every time step,

$$x_{t+1} = \begin{cases} 0.00001 & \text{if } x_t == 0 \text{ and } x_{desired} > 0 \\ x_t + k \cdot x_t, & \text{if } x_t < x_{desired} \\ x_t - k \cdot x_t, & \text{if } x_t > x_{desired} \\ x_t, & \text{otherwise} \end{cases} \tag{4.15}$$

where x_{t+1} is the output level in the next time step $t+1$; x_t is the current output level at time t; $x_{desired}$ is the desired output level; and k is a constant that determines rate of change in brightness. $x_{desired}$ must be between 0 and 1.

Formulating second and third cases of (4.15) as a first-order ODE shows that is an exponential function.

$$\frac{dx(t)}{dt} \pm k \cdot x(t) = 0 \tag{4.16}$$

Solving (4.16), we get

$$x(t) = x(0) \cdot e^k \cdot e^{\mp k \cdot t} \tag{4.17}$$

where $x(0)$ is the initial output level when the desired output level is changed.

This LED Driver enables the output level of the LED to increase at a faster rate in the higher brightness region.

4.5.3 Isolated CBLA Nodes

Each Isolated CBLA Node is associated with one actuator. A CBLA system, as discussed in Section 3.2, is constructed by linking these Isolated CBLA Nodes through virtual inputs.

In this test bed, three main types of Isolated CBLA Node exist: Half-Fin Node, Light Node, and Reflex Node.

Figure 4.10 presents the make-up of the different Isolated CBLA Nodes in a cluster. Two Half-Fin Nodes control the bending of a Fin through their respective SMA Controller Nodes (Fx.SMA-L and Fx.SMA-R). The pair of Half-Fin Nodes share a Fin-mounted IR proximity sensor (Fx.IR-F), and the 3 axes of the accelerometer (Fx.ACC). A Light Node controls the brightness of a high-power LED through an LED driver Node (Lx.LED) and its sensory space consists of an ambient light sensor (Lx.ALS). There are two types of Reflex Nodes, one is associated with a pair of LEDs (Fx.RFX-L), and one is associated with a vibration motor (Fx.RFX-M). They are both controlled through LED Driver Nodes. In their sensory space, they share one bottom-mounted IR proximity sensor (Fx.IR-S).

4.5.4 Network Configurations

The Isolated CBLA Nodes described in Section 4.5.3 are connected to each other via virtual inputs. In essence, the output of a CBLA Node is treated as an input variable to other Nodes, much like input variables associated with their own sensors. This creates a interconnected network of CBLA Nodes that spans across the entire sculptural system. This allows information regarding the external environment to travel through the sculpture.

Different network configurations produce different system behaviours. Here we investigate two types of network configurations which are called Spatial Mode and Random Mode. For fair comparisons, the two configurations have exactly the same number of links. In the experiment, the number of links was arbitrarily chosen to be 111 so that there are sufficient links to cover the entire system. In addition, each Node is linked to at least one other Node via virtual input.

Spatial Mode

In Spatial Mode, CBLA Nodes are linked based on their spatial proximity. Figure 4.11 shows the connections among Nodes within a cluster. Neighbouring Nodes are connected via bidirectional links. Two connected Nodes take each other's output as input variables. Nodes in neighbouring clusters that are in close proximity are linked via unidirectional links as shown in Figure 4.12. The circular connections that daisy-chain the four clusters allow information to spread throughout the test bed.

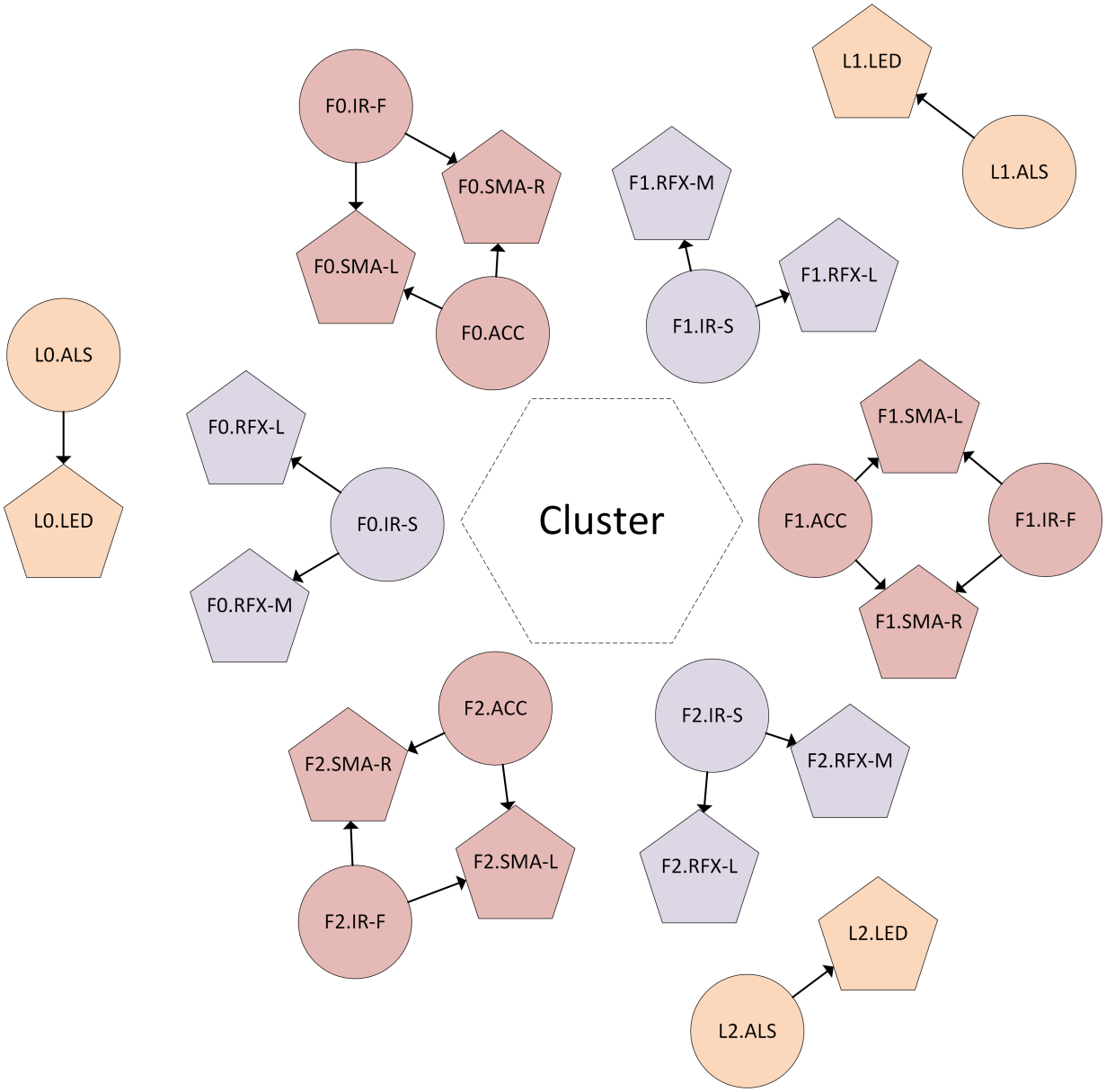


Figure 4.10: Make-up of a cluster of Isolated CBLA Nodes. Half-Fin Nodes are shown in red; Light Nodes are shown in orange; and Reflex Nodes are shown in blue.

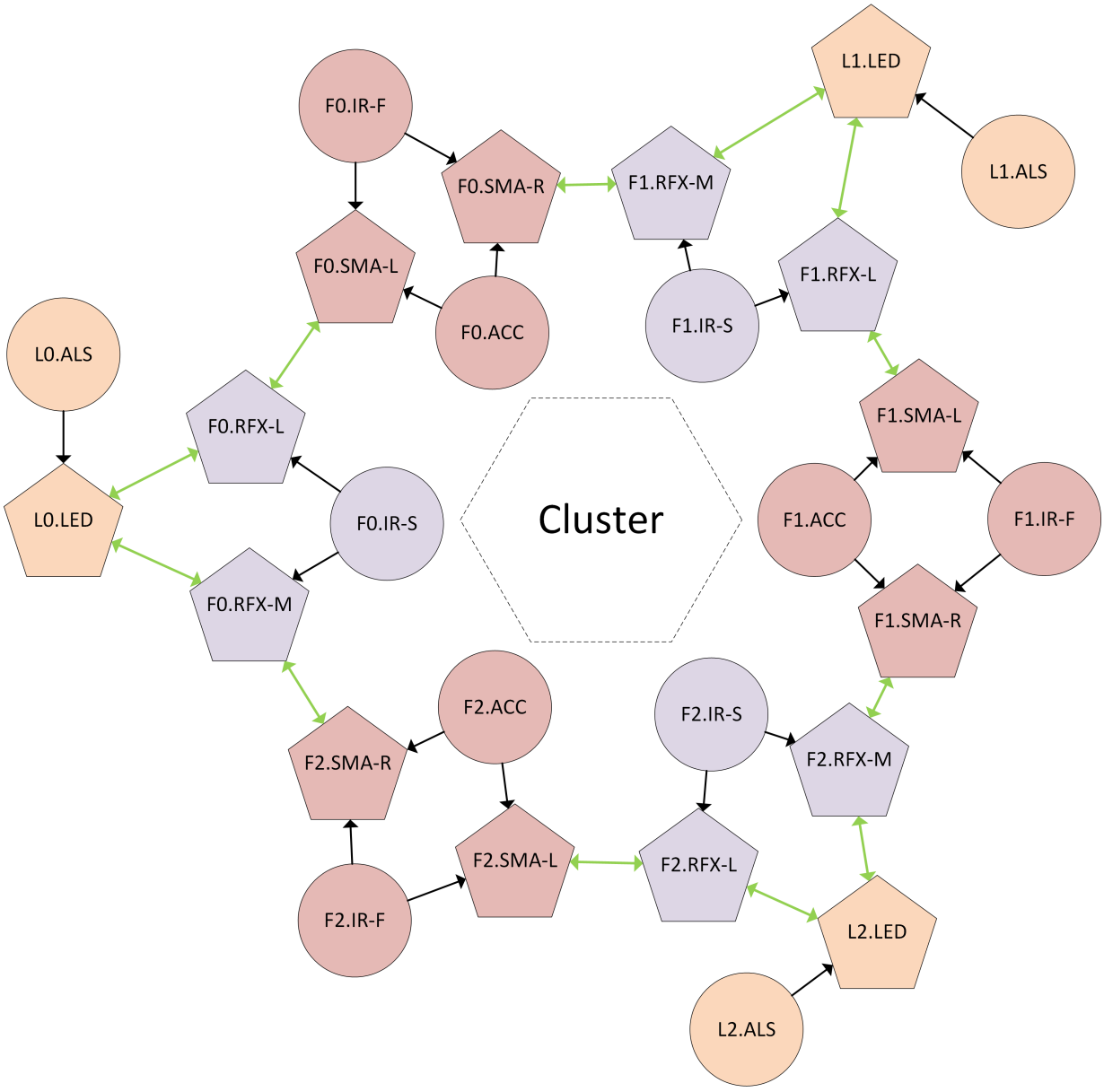


Figure 4.11: Connectivity graph within a cluster in Spatial Mode. Each bidirectional green arrow represents a pair of input links. Two connected Nodes take each other's output as input variable. Note that each bidirectional link is counted as two links.

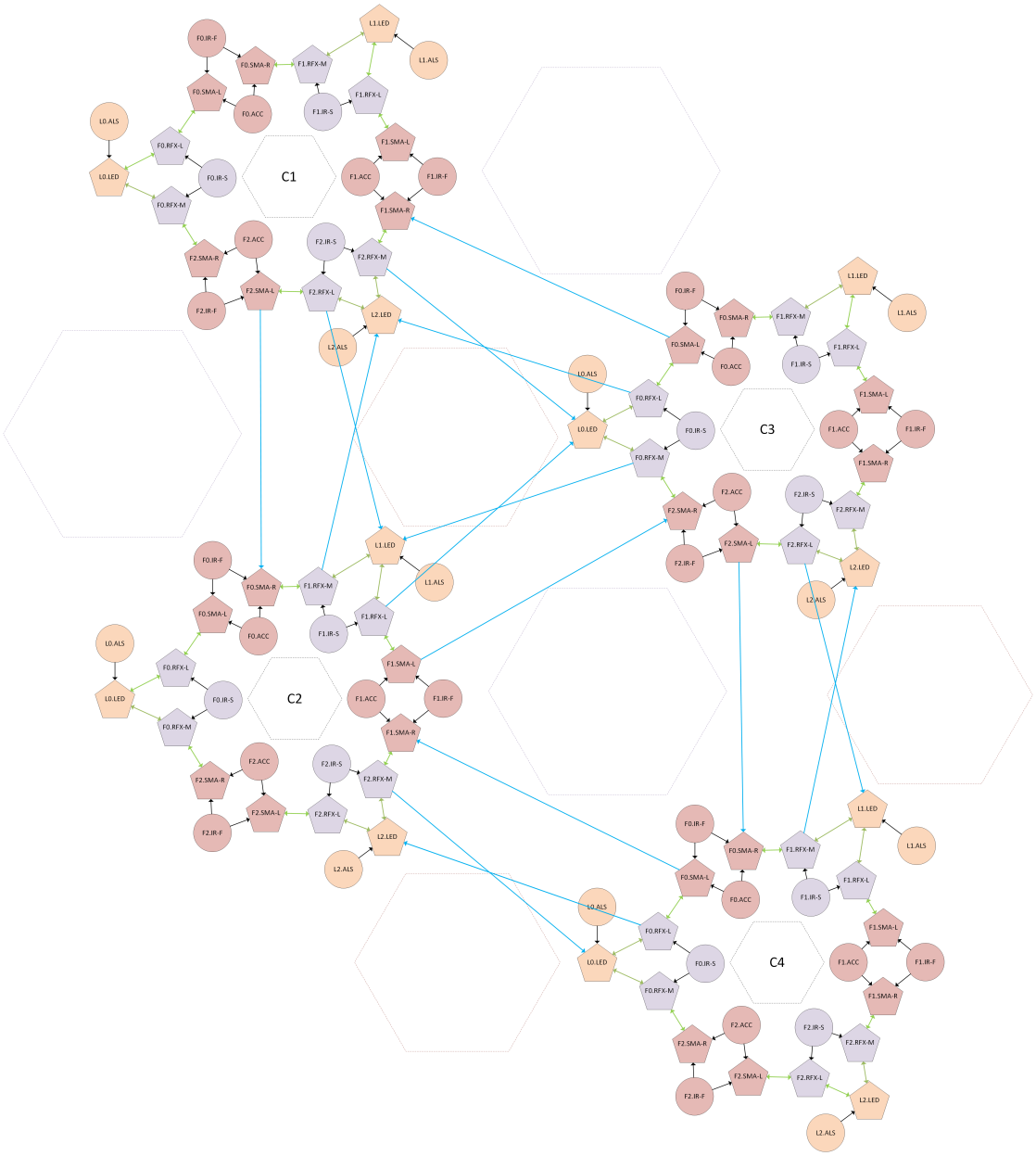


Figure 4.12: Connectivity graph of the entire test bed in Spatial Mode. Each blue unidirectional arrow represents an input link. The output of the Node at the origin of an arrow is fed into the Node that it is pointing toward as an input variable.

Under this configuration, Nodes' responses to external disturbances or environmental changes are expected to concentrate close to the source, and spread to other Nodes gradually.

Random Mode

In Random Mode, CBLA Nodes are linked to other CBLA Nodes randomly. This means that the output of a Node might be fed into another Node that is on the other side of the sculpture. Since there are 60 Nodes and 111 links are required, the algorithm first links the output of each Node to another random Node to ensure that each Node has an effect in the overall system. Then, the algorithm picks 51 unique Nodes at random and links each of them to another random Node. This ensures that the state space of each Node is relatively even.

The specific random configuration is generated at run-time and is different every time. In addition, it should be noted that this method does not guarantee that there will not be any disconnected sub-networks. It is possible that a set of Nodes is completely isolated from another set of Nodes under this network configuration scheme.

4.5.5 Prescripted Behaviours

For the purpose of comparing between CBLA and prescripted behaviours, each CBLA Node has a Prescripted Engine in addition to the CBLA Engine. This allows us to quickly switch between the two kinds of behaviours during the user study described in Section 5.4. Although the two engines are both associated with the same actuators, they may have different sensors in its sensory space.

The prescripted behaviours are implemented based on a specification by a human designer, and are similar to previous behaviours of the Hylozoic series[2]. For the Fin mechanism, when its Fin-mounted IR proximity sensor detects an object, it bends down toward the direction of a neighbouring bottom-mounted IR proximity sensor that has also detected an object. If both or neither of IR proximity sensors have detected an object, it simply bends straight down. It returns to an upright rest position when its Fin-mounted IR proximity sensor no longer detects an object in its proximity.

For the high-power LEDs, its output ramps up and down continuously when its corresponding Fin-mounted IR proximity sensor has detected an object. It then dims gradually when the object is removed.

For the reflex vibration motor or LED pair, its output also ramps up and down continuously when its corresponding bottom-mounted IR proximity sensor has detected an object. It then ramps down gradually when the object is removed.

An additional virtual node is added to provide cluster-level group behaviours. This node counts the number of outputs within its cluster that are active. It then determines a probability of random activation by mapping this count to a Gaussian function. The Fin mechanism or the high-power LED may turn on at random based on this probability. Using a Gaussian function allows the probability of random activation to increase when a number of outputs are activated. However, when too many outputs are activated, this probability decreases which makes random activations less probable.

Chapter 5

Experimental Validation

In this chapter, we demonstrate the behaviours generated by Curiosity-Based Learning Algorithm (CBLA) on an interactive art sculpture. We first investigated the behaviour of the simplest form of a CBLA system, one with a single node with one sensor and one actuator. This allows us to visualize the exploration pattern of the CBLA engine in two- or three-dimensional space. Then, we applied the algorithm on a small multi-node system with shared input variables. We observed its self-learning behaviours as well as the way it responds to external disturbances. After that, we constructed a small scale interactive sculptural system in the form of four-cluster test bed. In addition to shared inputs, virtual inputs were introduced to connect the different nodes. We investigated the different emergent behaviours resulting from different connection schemes. Finally, we conducted a formal user study using the test bed. Participants were invited to interact with the sculpture and report on their interest levels. The observations collected in this user study enabled us to understand the relationship between the participants' behaviours and engagement level under different conditions as well as different configurations of the CBLA system.

5.1 Single Node Experiment [1]

Although the CBLA was designed for a distributed system with multiple nodes, it might not be easy to visualize the modelling process due to the high-dimensionality of the data and the models. To demonstrate the action selection pattern and the learning process, the

[1] An early version of this section has been published at IROS 2015 [1]

CBLA was first tested on a simple toy example which is easily visualizable in 3-dimensional space. In this experiment, idle mode was disabled as the main objective was to observe and verify the exploration pattern of the CBLA.

5.1.1 Set-up

The system in this experiment consists of a Light node, which is a single-input, single-output system. For the single-input system, S is a scalar value that represents the measurement from an ambient light sensor. It was recorded directly as a 12-bit value. M corresponds to the voltage duty cycle supplied to the LED, ranging from 0 to 100, with 0 being completely off (0V) and 100 being the maximum allowable voltage (4.7V). The loop period is the time between each actuation and was set to 0.05s.

5.1.2 Procedures and Expected Results

In this experiment, the system ran for 2500 time steps without any external interference. Based on the reward structure, which favours learning first the most predictable regions of the state-space, the CBLA is expected to first explore the regions of the sensorimotor space that have low variance. Once the model in that region is learnt, it should move onto areas with higher variance.

5.1.3 Results

Figure 5.1 shows the evolution of the prediction model and actual exemplars over time. As expected, the CBLA first selects actions associated with lower LED output levels, as this leads to measurements in the low variance regions. Over time, once the model in the low variance region is acquired, it moves toward higher brightness regions. Figure 5.2 shows that the best action and the actual selected action were completely random at first. The system then focused on the easy-to-learn areas in the lower brightness level. After that, it moved toward the higher brightness and harder-to-learn regions when it hadn't seen much improvement in the low brightness regions. After some exploration of the bright regions, prediction error is reduced in those regions, and the system returns again to explore the low-brightness region. The resulting pattern of activation is interesting visually, as it results in non-random activations that have the potential to convey a notion of intent to the viewer.

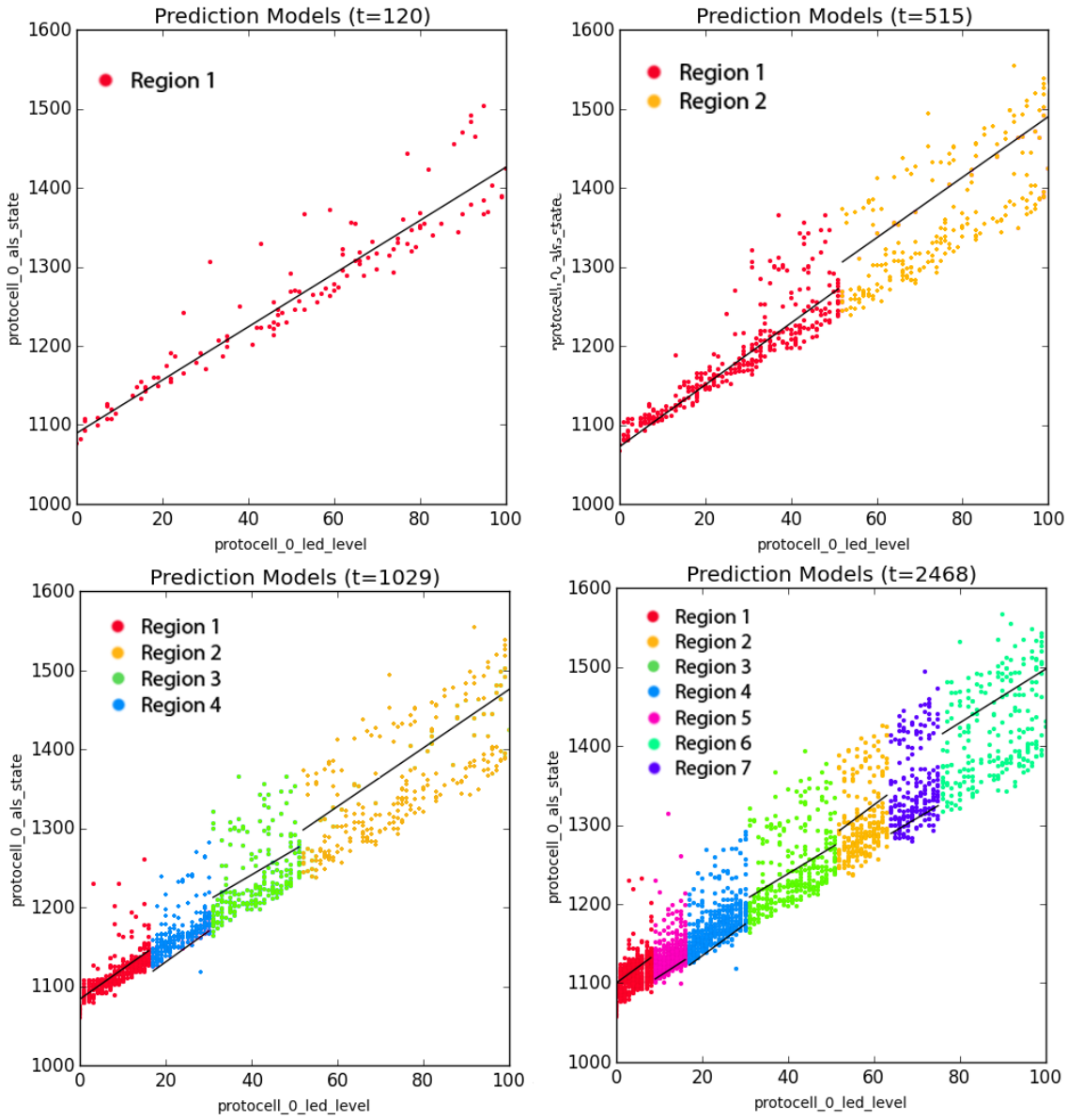


Figure 5.1: Evolution of the prediction models for the single node experiment. Each point represents an exemplar. Points with the same colour are in the same region and the black lines are the cross-section of the linear models at S(t) = 0. The regions are numbered in the order that they were created.

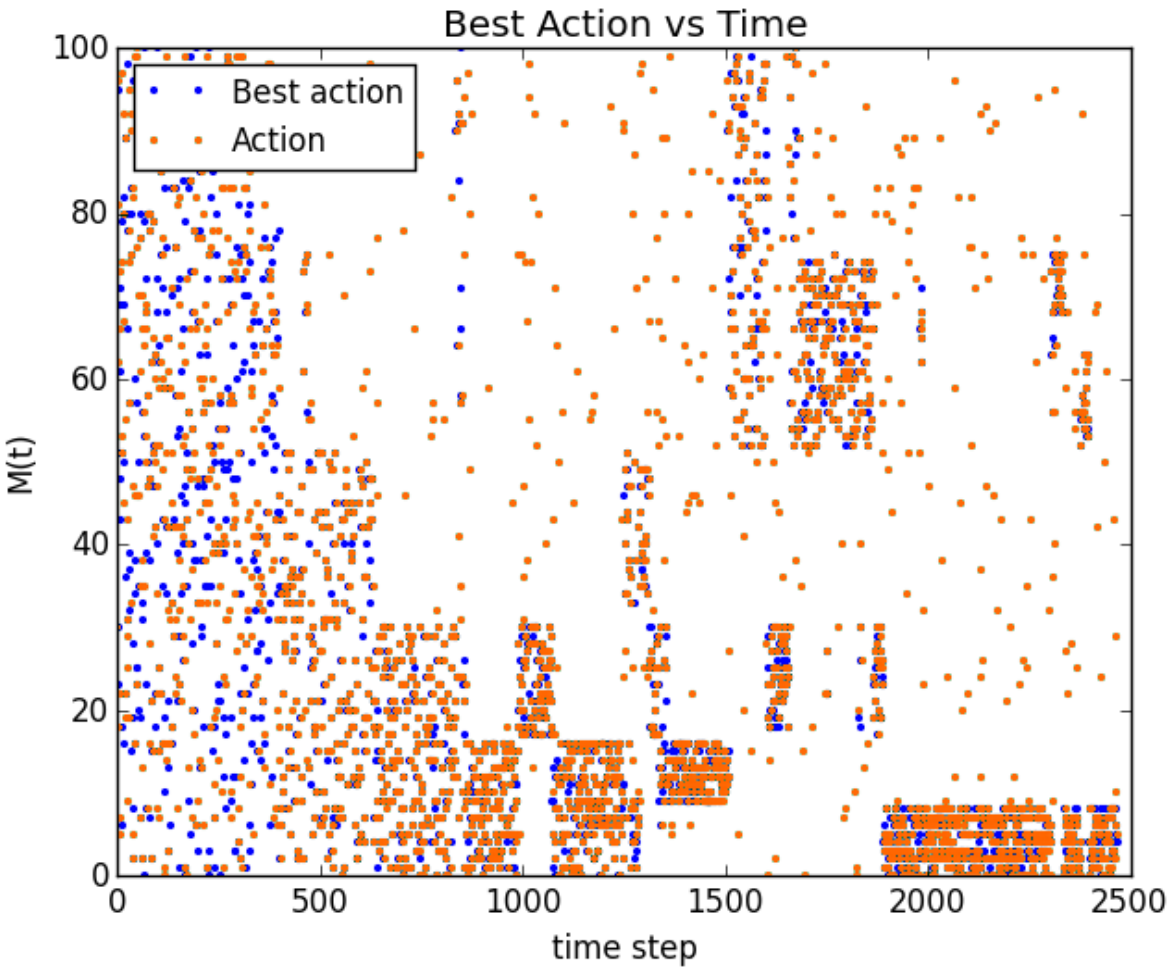


Figure 5.2: Action vs Time Graph for the single node experiment; the y-axis is the output of the LED M(t) and the x-axis is the time step. Orange dots represent the actual action taken and blue dots represent the best action given the sensorimotor context. The best action is defined as the action with the highest action value given the current state. Non-best actions are selected occasionally in order to explore the state space.

Figure 5.3 shows the mean error vs. time graph. Here we see that the prediction error quickly drops to a relatively low level. To improve its prediction further, the state-space was split into regions with low and high error. This allows the Region 1 (low variance region) to further reduce its prediction error.

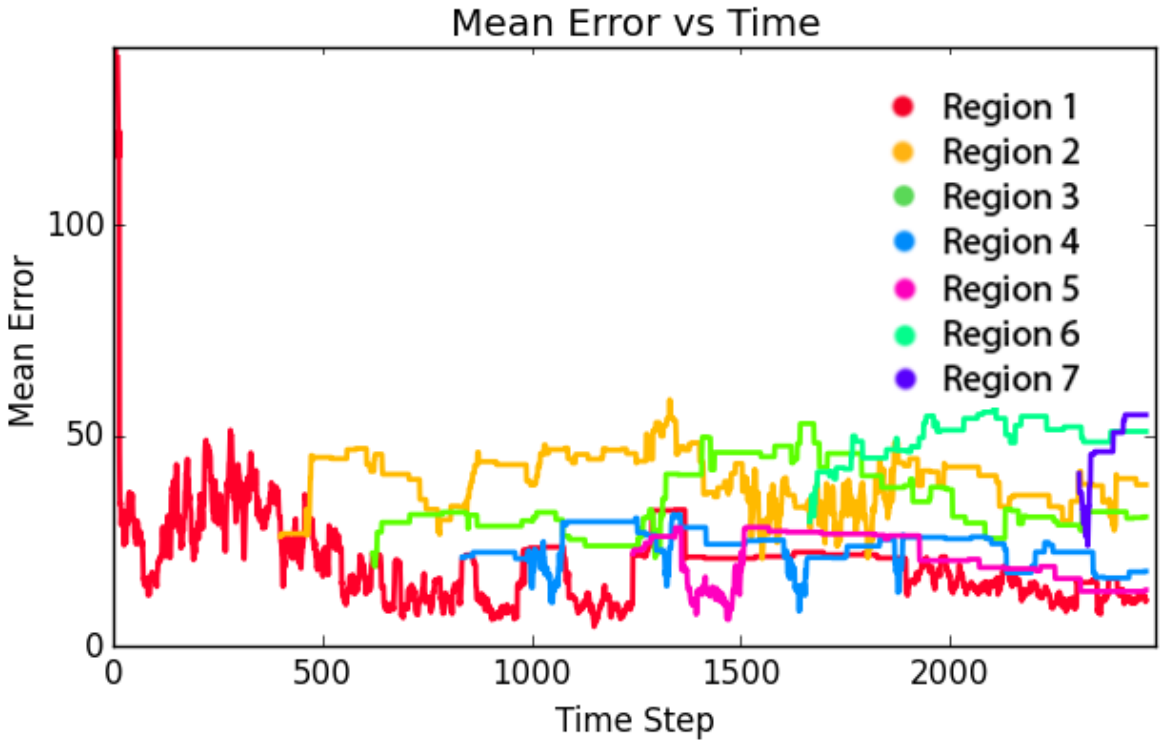


Figure 5.3: Mean error vs time graph for the single node experiment. Each colour represents a region and the colour code corresponds to final prediction model graph in Figure 5.1

In Figure 5.4, one can that see the action value of a region does not stay constant. This shows that as the prediction improves, the value of actions in that region decreases over time as the region becomes learnt and further learning potential decreases.

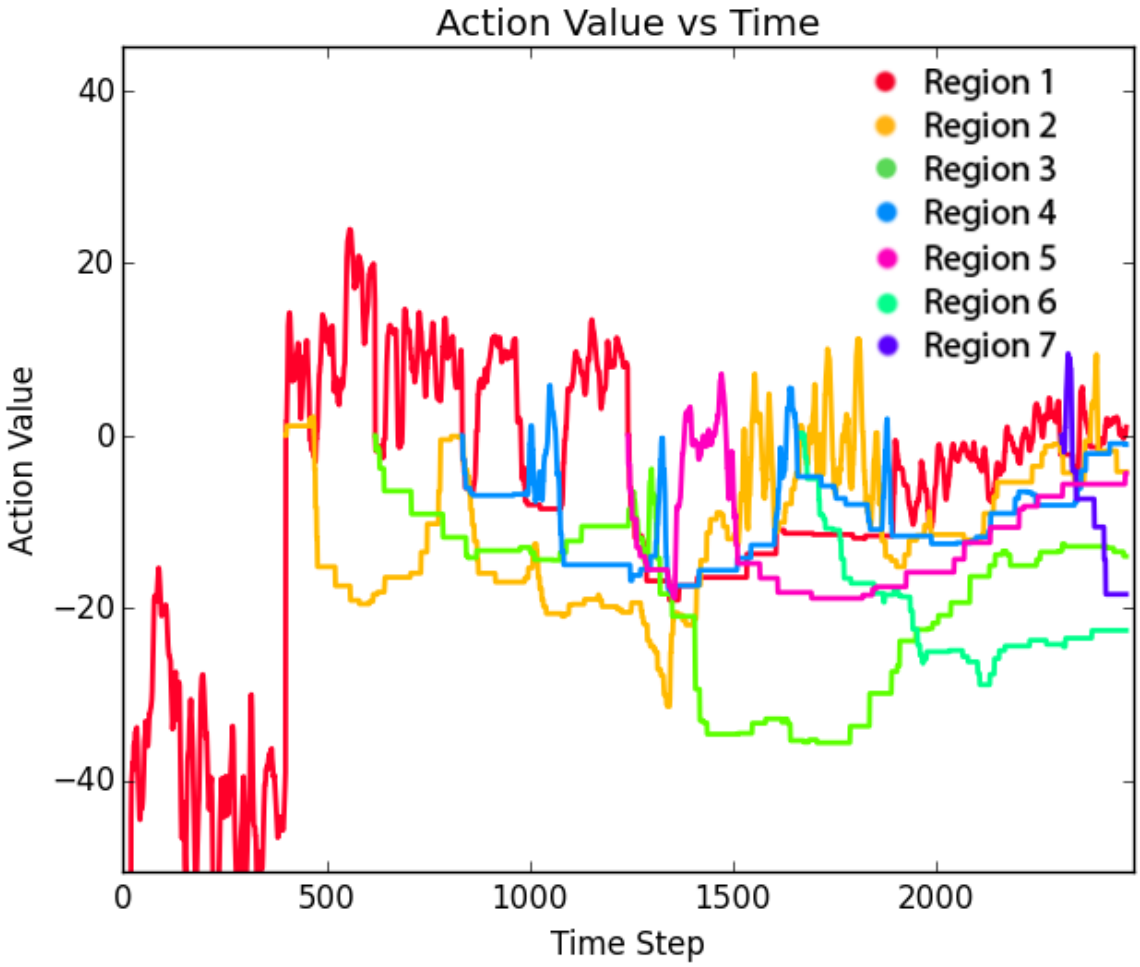


Figure 5.4: Action value vs time graph for the single node experiment. Each colour represents a region and the colour code corresponds to final prediction model graph in Figure 5.1

5.2 Multi-Node Experiment [2]

In this Section, we describe a demonstration of an integrated system consisting of multiple CBLA Nodes. In addition, in this experiment, threshold-based Idle Mode was introduced. When the knowledge gain potential is low, a Node would enter Idle Mode and turn off its actuators. The behaviours of the system during the self learning period and its response to external interference was examined.

5.2.1 Set-up

The Light node was the same as in Section 5.1, with the addition of the shared IR proximity sensor. For the Fin node, the input variables are the average accelerometer readings of the three axes, and the shared IR proximity sensor reading over the 12.5s loop period; the output variable is the action of the Fin. There are four discrete actions: rest (0), lower to the right (1), lower to the left (2), and lower to the centre (3). Note that in this set up, the two types of nodes run with different loop periods, but coupling between them is accomplished through the shared IR sensor, which measures proximity as a 12-bit value. Note that in this experiment all four nodes share one single IR sensor. The set-up of the experiment is shown in Figure 5.5.

5.2.2 Procedures and Expected Results

The system runs undisturbed until, after some initial learning, all of the nodes enter Idle Mode. During this time, the IR proximity sensor pointed toward an empty area. Afterwards, a participant enters into the sculpture space in an area detectable by the IR proximity sensor. The system should then exit idle mode and begin learning the changed model introduced by the change in the environment. Since the IR sensor is shared by all nodes, they are all expected to recognize the change and exit idle mode at approximately the same time.

5.2.3 Results

Figure 5.6 shows how the action values change over time for each of the nodes. The coloured lines represent the action values and each colour represents a region. The blue dots underneath the plot indicate when the node was in idle mode.

[2] An early version of this section has been published at IROS 2015 [1]

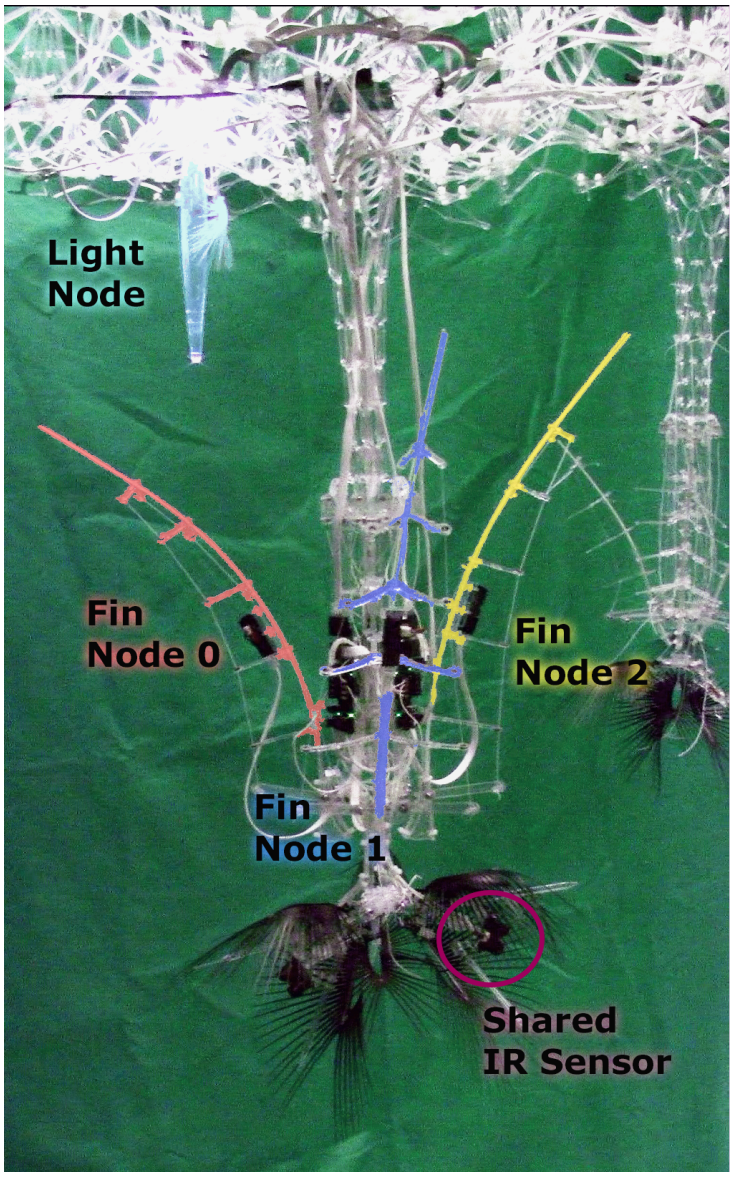


Figure 5.5: The set-up of the Multi-Node Experiment. The Light Node is shaded light blue, and the three Fin Nodes are shaded red, blue, and yellow. All four nodes share an IR proximity sensor in the purple circle.

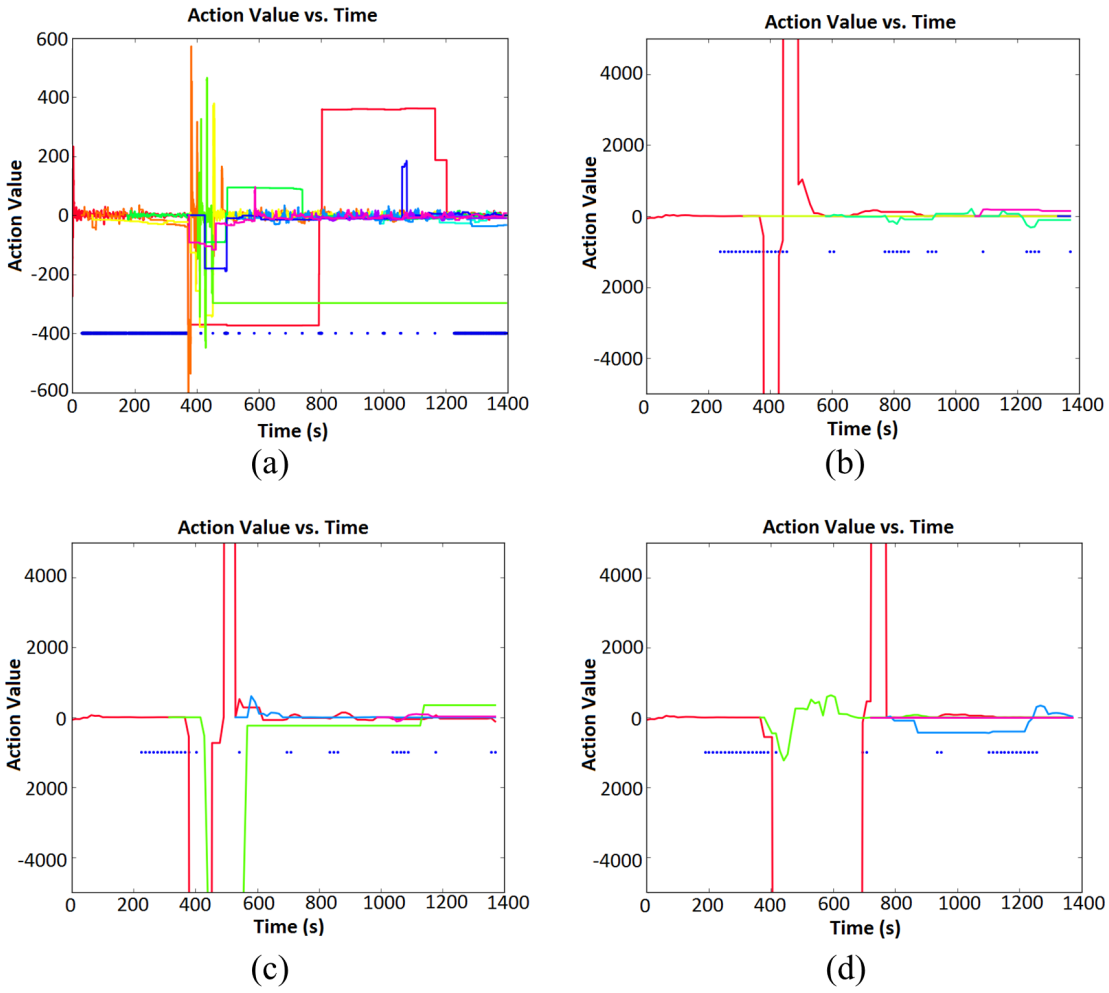


Figure 5.6: Action-value vs Time graph for the Light node (a) and the three Fin nodes (b), (c), (d).

All the nodes first started learning their own models and entered idle mode. At around 390s, a human participant walked in front of the IR proximity sensor. This triggered a large reduction in action value at first, due to an increase in prediction error. However, as more data was collected, the action values for all four nodes quickly jumped up. This prompted the nodes to exit idle mode and begin generating actions to learn the model. After a period of readjustment, all four nodes re-entered idle mode after the new environment is learnt.

5.2.4 Discussion

From Figure 5.6, one can see that the Light Node and the three Fin Nodes reacted to the environmental change nearly simultaneously. They exited the idle state and shifted to more exploratory actuation patterns. This showed that the shared sensor input variable was able link the CBLA engines together, even though they run independently at different frequencies. This experiment demonstrates that the reaction of the system to the changed environmental conditions creates an interaction with the visitor without any explicitly pre-programmed behaviours. The system appears to respond to the participant and their action, as its internal model of the environment cannot predict the behaviour of the participant.

The system's intrinsic curiosity drives itself to perform actions and elicit responses from this new environment with the participant's presence, and update its prediction model. We anticipate that the visitors will find such behaviours engaging as the visitors can recognize that the sculpture is responding to their presence and action but they would not be able to easily predict how it might respond. This quality provides the CBLA System the potential to be more life-like than a prescripted or a random system.

5.3 Multi-Cluster Experiment

This experiment investigates the behaviours of the CBLA system on a four-cluster test bed described in Section 4.5. The test bed has two different network configurations: Spatial Mode, and Random Mode. Under different network configurations, the CBLA system is expected to behave differently. We hypothesize that users would find activations closer to them more relevant, and hence more interesting. Therefore, a metric to quantify this proximal activation is devised. The configuration with higher proximal activation is then used for subsequent user study described in Section 5.4.

5.3.1 Set-up

The multi-cluster test bed described in Section 4.5 is used for this experiment. Since the goal of this experiment is to examine the behaviours of the CBLA system, the Prescripted Engine is not used in this experiment. In addition, within the CBLA Engine, a sigmoid function based idling function as described in Equation (3.5) is used.

5.3.2 Procedures

The CBLA system first starts from a blank state without any preexisting exemplars or prediction models. It operates without any external interference for 300 seconds. After that an object is placed in front of the IR proximity sensor C3.F2.IR-S (referring to Figure 4.12) for approximately 30 seconds. Then, the object is removed and the experiment continues on for another 120 seconds before the program is terminated.

This procedure was repeated three times for both Spatial Mode and Random Mode.

5.3.3 Expected Results

The level of output by an actuator is referred to as activation. All clusters are expected to have approximately the same level of activation during the initial learning period. After that, the amount of activation would reduce as the knowledge gain potential decreases.

For Spatial Mode, it is expected that the activation as a result of the trigger at 300s point should be concentrated around the external interference location, which is near Cluster 3 ($C3$). On the other hand, for Random Mode, since Nodes are connected randomly, activation should be more spread out and no one cluster should have a larger than average level of activation. To quantify the activation level, metrics that evaluate the average total activation and average cluster activation are devised.

The total activation level at time step t is the average output levels of all CBLA Nodes at time step t. It can be calculated as follows.

$$a_t = \frac{1}{N} \sum_{j=0}^{N} \overline{m}_{jt}, \text{ for all Nodes } j \tag{5.1}$$

where a_t is the total activation level at time step t; N is the total number of CBLA Nodes; $\overline{m}_{jt}$ is the average output value of CBLA Node j at time step t.

We are interested in the total activation level during the period immediately after the trigger was applied. Therefore, we compute the average total activation between the time of the trigger, t_{trig}, and the end of the readjustment period, t_{readj}, as follows.

$$\overline{a} = \frac{1}{|\left\{a_{T_{trig}}, ..., a_{T_{readj}}\right\}|} \sum_{t=T_{trig}}^{T_{readj}} a_t \tag{5.2}$$

where $\overline{a}$ is the average total activation from T_{trig} to T_{readj}; and $|\left\{a_{T_{trig}}, ..., a_{T_{readj}}\right\}|$ is the number of time steps between T_{trig} and T_{readj}.

By looking at the cluster activation level, we can determine if the activations are concentrated in any particular cluster. To find the cluster activation level, we apply (5.1), but only for the CBLA Nodes within a particular cluster.

$$a_{ct} = \frac{1}{N_c} \sum_{j=0}^{N_c} \overline{m_{jt}}, \text{ for all Nodes } j \text{ in cluster } c \tag{5.3}$$

where a_{ct} is the cluster activation level at time step t; N_c is the total number of CBLA Nodes in cluster c; $\overline{m}_{jt}$ is the average output value of CBLA Node j at time step t.

Similarly, we can calculate the average cluster activation level during the period between t_{trig} and t_{readj} as follows.

$$\overline{a_c} = \frac{1}{|\left\{a_{cT_{trig}}, ..., a_{cT_{readj}}\right\}|} \sum_{t=T_{trig}}^{T_{readj}} a_{ct} \tag{5.4}$$

where $\overline{a_c}$ is the average cluster activation from T_{trig} to T_{readj}; and $|\left\{a_{cT_{trig}}, ..., a_{cT_{readj}}\right\}|$ is the number of time steps between T_{trig} and T_{readj}.

In this experiment, we set T_{readj} to 360s. This means that we are looking at the behaviours of the system within 60 seconds since the trigger at $T_{trig} = 300$s. This time interval was chosen as we wanted to investigate how the responses of CBLA system in the relatively short time duration after users introduce the disturbances. We assumed that the users would not associate their actions with the activations that appear after more than 60s have passed.

5.3.4 Results

The average total activation and average cluster activation values for the Spatial and Random Mode trials are listed in Tables 5.1 and 5.2. The means of the results from each set of the three trials were used to interpret the results.

Table 5.1: Results of the multi-cluster experiments for Spatial Mode. The average total activation ($\overline{a}$) and the average cluster activation for each cluster ($\overline{a_{c1}}$, $\overline{a_{c2}}$, $\overline{a_{c3}}$, and $\overline{a_{c4}}$) were computed from $T_{trig} = 300$s to $T_{readj} = 360$s.

Trial #	$\overline{a}$	$\overline{a_{c1}}$	$\overline{a_{c2}}$	$\overline{a_{c3}}$	$\overline{a_{c4}}$
1	0.07209	0.06259	0.07982	0.09966	0.04629
2	0.07306	0.08146	0.07130	0.08726	0.05219
3	0.06915	0.08628	0.04721	0.08548	0.05761
Mean	0.07143	0.07678	0.06611	0.09080	0.05203
Std. Dev.	0.00203	0.01251	0.01691	0.00772	0.00565

Table 5.2: Results of the multi-cluster experiments for Random Mode. The average total activation ($\overline{a}$) and the average cluster activation for each cluster ($\overline{a_{c1}}$, $\overline{a_{c2}}$, $\overline{a_{c3}}$, and $\overline{a_{c4}}$) were computed from $T_{trig} = 300$s to $T_{readj} = 360$s.

Trial #	$\overline{a}$	$\overline{a_{c1}}$	$\overline{a_{c2}}$	$\overline{a_{c3}}$	$\overline{a_{c4}}$
4	0.03297	0.04193	0.02686	0.02951	0.03358
5	0.08062	0.08114	0.07884	0.08035	0.08218
6	0.05953	0.06105	0.05732	0.06237	0.05737
Mean	0.07008	0.06137	0.05434	0.05741	0.05771
Std. Dev.	0.01491	0.01960	0.02611	0.02578	0.02429

For both Spatial mode (Trial 1, 2, and 3) and Random Mode (Trial 4, 5, and 6), their average total activation values, $\overline{a}$, were similar. However, the variance is much greater for Random Mode. The level of variation may be attributed to the fact that the links connecting its CBLA Nodes were different at every trial.

On the other hand, for Spatial Mode, the average cluster activation values for Cluster 3 is larger than the other clusters. Cluster 3 is where the trigger took place in the experiment. While for Random Mode, the values of different clusters are relatively consistent. This shows that activations do tend to concentrate more around the origin of the trigger in Spatial Mode in contrast to Random Mode as we expected.

The total activation and the cluster activations over time for Trial 2 (Spatial Mode) are shown in Figure 5.7. At the point at $t = 300$s when the object was presented in front

of a sensor in Cluster 3, the activation level at the cluster (red) shot up and the activation levels in the other three clusters trail after as we expected. Furthermore, it is interesting to point out that the magnitudes of activations for the other clusters are actually similar or even higher than Cluster 3. This means that CBLA Nodes in a cluster would respond to an event happening in a neighbouring cluster with similar level of intensity while the timing may be delayed.

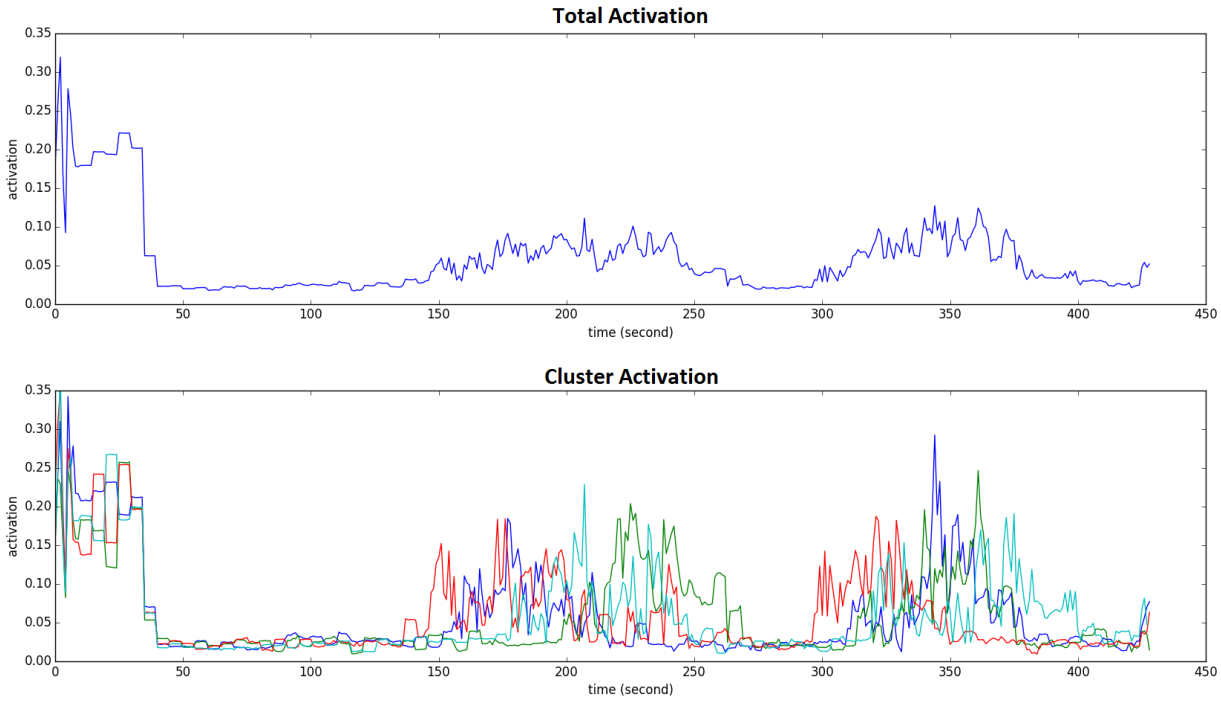


Figure 5.7: Average total activation and average cluster activations for Trial 2 (Spatial Mode) of the multi-cluster experiment. In the Cluster Activation plot (bottom), the blue line represents Cluster 1; the green line represents Cluster 2; the red line represents Cluster 3; and the cyan line represents Cluster 4.

In contrast, in the total activation and the cluster activations over time plots for Trial 5 (Random Mode) shown in Figure 5.8, the activation levels across different clusters are relatively even and without much delays. This is also expected since the CBLA Nodes are linked randomly, there is no expectation that CBLA Nodes in one cluster should activate earlier than another.

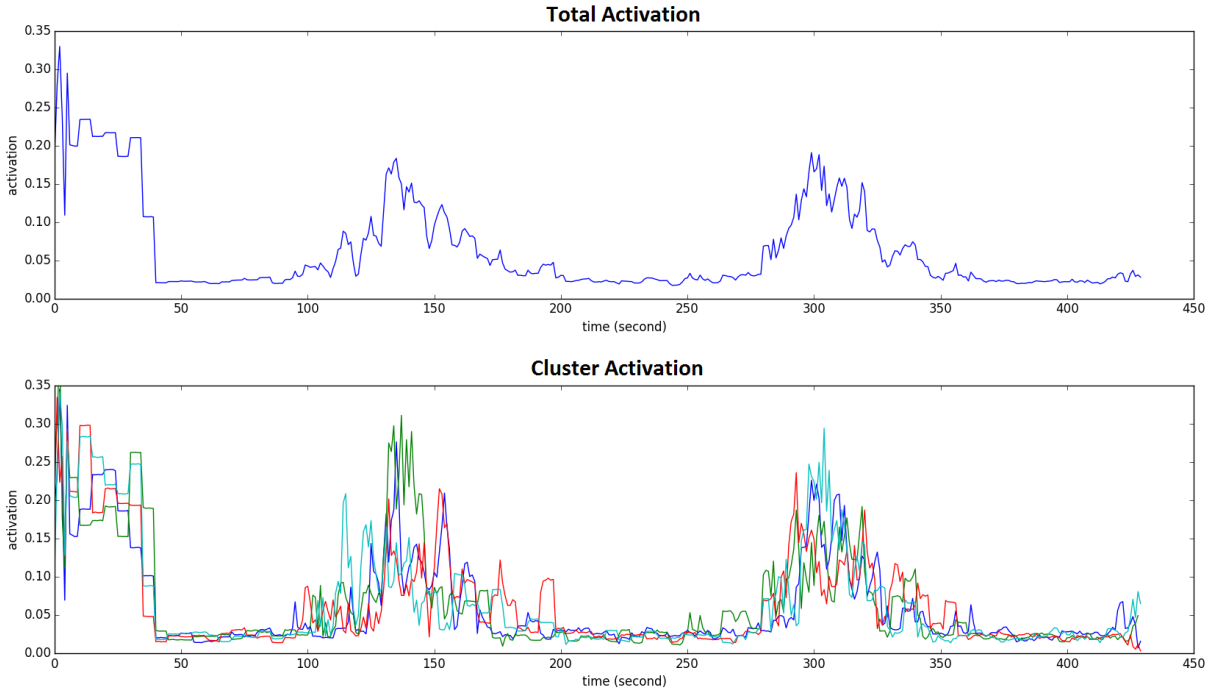


Figure 5.8: Average total activation and average cluster activations over time plots for Trial 5 (Random Mode) of the multi-cluster experiment. In the Cluster Activation plot (bottom), the blue line represents Cluster 1; the green line represents Cluster 2; the red line represents Cluster 3; and the cyan line represents Cluster 4.

5.3.5 Discussion

In this experiment, we showed that the activations in response to a triggering event are more likely to begin and concentrate near the source of the trigger when the network is configured as Spatial Mode. Since we hypothesize that the users would find activations in close proximity to be more relevant and interesting, Spatial Mode seems to be a more promising network configuration. Therefore, in the User Study described in Section 5.4, the CBLA system was configured in Spatial Mode.

Another observation made in the experiment was that there seems to be some periodic spontaneous activations for either network configuration. For example, in both Figures 5.7 and 5.8, there were activation at around 100s to 200s points, even though there was not any external interference. In fact, over a longer period of time, a self activation pattern becomes evident as shown in Figure 5.9 where the system was run in Spatial mode for

4500s uninterrupted. The CBLA Nodes activated spontaneously at a relatively consistent rate despite the absence of any external triggering events.

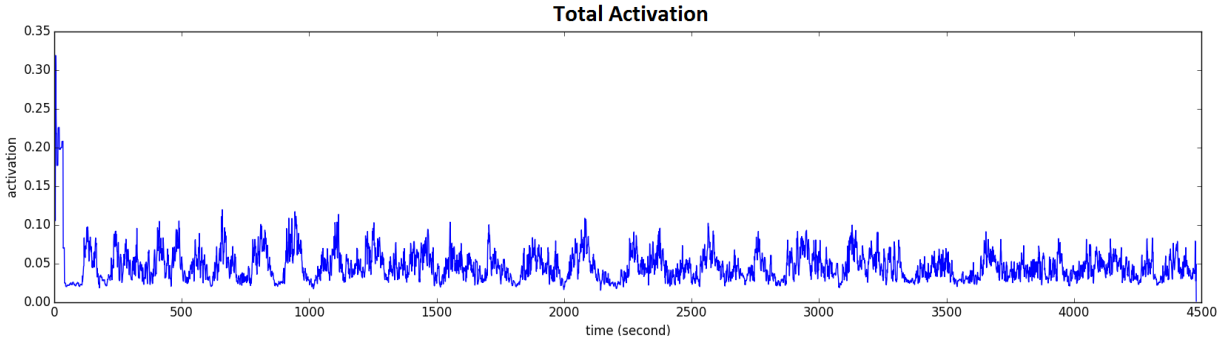


Figure 5.9: The CBLA system was running uninterrupted for 4500s in Spatial Mode. Even without any external events, the CBLA Nodes activated at a relatively consistent rate.

As mentioned in Section 3.1.5, this periodic activation pattern is caused by the way which the relative action value, $\widehat{q_z^2}$, is calculated. Looking back at Equation (3.8),

$$\widehat{q_z^2} = \frac{q_t^2}{max(\overline{q_z^2}, \nu)}$$

we can see that when the average squared action value, $\overline{q_z^2}$, hovers close to 0 for a long time, relative action value, $\widehat{q_z^2}$, would become very large. Fundamentally, this pattern is caused by the system's sensitivity to ambient sensor noise and can be adjusted by changing the sensitivity constant, ν. Although setting ν to a larger value can eliminate this periodic activation pattern, it would also make the sculpture less responsive to the users and to the external environment. In addition, we believe that this kind of patterns can indeed be an interesting aspect of the behaviours of this interactive art sculpture.

5.4 User Study

The CBLA system is designed to automatically generate interactive behaviours on interactive art sculptures. This user study aimed to determine whether the behaviours generated through this method can make the experience of interacting with the sculpture more interesting, compared with prescripted behaviours.

In this study, the test subjects reported their levels of interest at several points in time as they interacted with sculpture, which had two versions of behaviours. Afterwards, a

short exit questionnaire was given to assess the subjects' overall experience. The results of this study enables designers to design more engaging and interesting interactive art sculptures.

5.4.1 Objectives

This study aimed to investigate the users' responses to the behaviour of the CBLA system in an interactive art sculpture under different configurations. In addition, the relationships between the intensities and types of activations and users' level of interest as they interact with the sculpture were examined.

The user study aimed to answer the following three questions.

1. Does the use of the CBLA increase user's interest level over prescripted behaviours?
2. Do people perceive CBLA as non-random?
3. Are certain behaviours more interesting than others?

Hypotheses

Prior to the user study, an hypothesis was made for each of the research questions raised.

1. The CBLA works by continuously generating new behaviours in order to improve its internal mathematical model of the sculpture and its sensed environment. The behaviours are adaptive and analogous to how animals and human beings learn. The users will find this kind of behaviour more interesting than prescripted behaviours.
2. Although the CBLA continuously generates new behaviours, it is not random. the users will not perceive the CBLA-generated behaviours as random.
3. The users categorize some types of behaviours as being more interesting than others.

Figure 5.10: Photograph of the floor grid underneath the multi-cluster test bed.

5.4.2 Set-up

The multi-cluster test bed described in Section 4.5 was used in this user study as the interactive art sculpture that the participants interact with. In addition, the floor was lined with a grid numbered from 1 to 12 as shown in Figure 5.10. This grid allowed the participants to specify their locations within the sculpture.

In addition, a screen was set up next to sculpture to display the sample number. A laptop running the software was set up facing away from the sculpture beside the screen.

5.4.3 Recruitment

Participants were recruited from the researchers' contact lists. All participants were healthy, physically able adults within the age range of 18 to 65 years old, male and female. Potential participants who were blind were excluded because they would be unable to perceive many of the behaviours, which were visually-manifested. Potential participants who were wheelchair users were excluded because they would not be able to access the studio, which had no elevator access.

In addition, participants had little prior knowledge about that the workings of the CBLA system to avoid the subject-expectancy effect. This is a form of bias in which the test subject expects a particular result and this unconsciously affects the outcome.

5.4.4 Procedures

Each participant was provided with the same information about the procedures of this user study. Only one participant was interacting with the sculpture at a time.

Before the Trial

The test participant was invited to interact with an interactive art sculpture installed in the Toronto studio of Philip Beesley Architect Inc. (PBAI). The participant was then informed about the procedures of the study as described in the Information and Consent Form (Appendix A.1) and was asked to sign the attached consent form before he or she began participating in the trial.

After that, the participant was given a pen and an envelope with a stack of 8 identical business-card-sized questionnaire cards shown in Figure 5.11. He or she was asked to fill

out one card every time a long tone was heard. The first question on the card asks the subject to write down the current sample number as shown on the screen. The second asks for his or her subjective interest level regarding the behaviour of the sculpture at that moment. The third question asks the subject to mark his or her location at that moment by looking down on to the grid on the floor. He or she was told to make a mark between the boxes if he or she was standing in between those boxes.

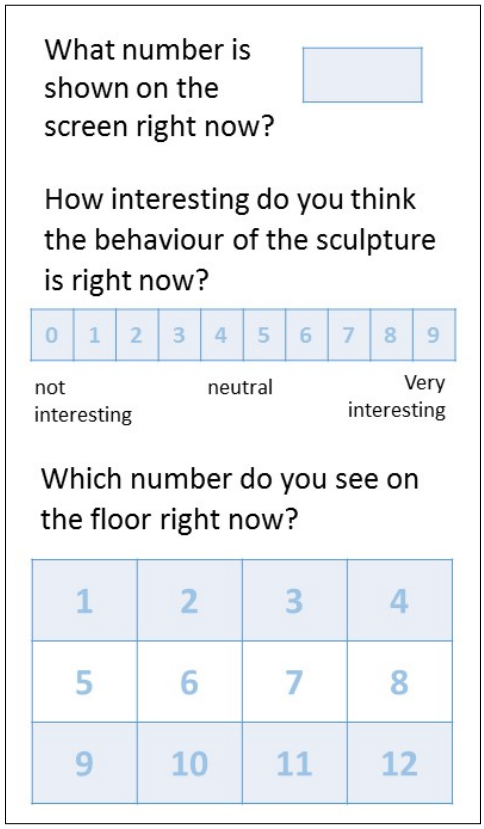


Figure 5.11: Questionnaire card for the user study.

Right before the start of the trial, the researcher ran through the study procedures with the participant one last time. The participant was told that he or she was free to walk around the space and interact with the art sculpture in any way and that there were not any prescribed interactions. The subject was also told that the entire trial would last for 20 minutes and he or she may request to terminate the study at any time.

During the trial

The test participant was free to roam around the space and interact with the art sculpture. A distinct long tone went off periodically at a 2.5 minutes interval. Each time the tone went off, the participant would take out an empty questionnaire card, and answer the questions on the card. The trial would go on for 20 minutes which was equivalent to 8 cards.

There were two versions of interactive behaviours: Prescripted Mode and CBLA Mode. The Prescripted Mode was powered by the Prescripted Engine and is described in Section 4.5.5. The CBLA Mode was powered by the CBLA Engine and is described in detail in Chapter 3. The participants were not informed about the versions of the behaviours that they were interacting with nor the fact that there were two different versions of interactive behaviours. Both versions of the behaviours were run for approximately the same amount of time. After the fourth tone (which was half-way through the trial), the researcher would switch to the other version manually after the participant had finished filling in the questionnaire card and resumed interacting with the sculpture.

Since the experience of interacting with the sculpture was new to the test subjects, the order that the two types of behaviours was presented can make a big difference. The subjects may simply find the novelty of interacting with the sculpture interesting irrespective of the type of behaviour. In addition, since the exit questionnaires were given after the trials, the behaviours that the subjects experienced near the end of the trials may have more profound effects than the ones that they experienced earlier in the trials. Therefore, half of the participants were exposed to Prescripted Mode first and the other half were exposed to CBLA Mode first.

After the trial

After a participant had filled out the eighth questionnaire card, he or she was informed that the trial was completed. He or she then returned the questionnaire cards in the envelope given to him or her prior to the trial. The participant then filled out an exit questionnaire, shown in Appendix A.2, which has the following five questions.

1. How interesting was your overall experience while interacting with the art sculpture?

2. How responsive was the sculpture to your presence? Do you think its behaviour was totally random, or do you think it was responding directly to you?

3. How familiar are you with machine learning algorithms?

4. How would you describe the behaviour of the sculpture?

5. Additional comments

Question 1 elicits the participant's general opinions about how interesting it was interacting with the sculpture. Question 2 can provide insights on research question 2. Both questions 1 and 2 were rated on a scale from 0 to 9. Question 3 allows us to gauge whether or not the participant's prior exposure to machine learning had any effect of his or her perceptions of the behaviours of sculpture and it was rated on a scale from 0 to 4. Questions 4 and 5 are open-ended questions that can give us some qualitative information about the participant's experience that we might not capture otherwise.

After that, the participant was provided with the debriefing letter shown in Appendix A.3, which has more information about the learning algorithm, the interactive behaviours, and the purpose of this study. In addition, the interactive behaviours that he or she interacted with were explained verbally after all questionnaires were collected.

5.4.5 Results and Data

A total of 10 participants were recruited. There were three main sets of data collected in each trial: the set of 8 questionnaire cards, the exit questionnaire, and the states of the CBLA system during the entire trial.

The questionnaire cards recorded each subject's levels of interest and locations at the 8 sample points. The levels of interest data enabled us to compute the subject's overall interest level in each of the two segments, Prescripted Mode and CBLA Mode. We further separated the data into two groups based on the trial started with Prescripted Mode first or CBLA Mode first and they are presented in Tables 5.3 and 5.4. On the other hand, the location data enabled us to examine the relationships between the activations near the test subject to his or her reported interest level.

The exit questionnaires provided information about the participants and their overall opinions and thoughts on their experience before any additional information about the user study was revealed. In addition, observations of the participants' behaviours and interesting comments made in verbal conversations with were noted to provide further insights.

Table 5.3: Self-reported interest levels at the sample points collected during Prescripted-first trials in the user study

Trial #	**Sample Interest Level**							
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**
1	6	4	6	3	7	6	4	7
3	1	4	2	3	5	3	3	2
5	4	4	5	5	3	3	5	4
7	7	6	7	5	5	6	5	6.5
10	9	9	7	5	5	3	3	2

Table 5.4: Self-reported interest levels at the sample points collected during CBLA-first trials in the user study

Trial #	**Sample Interest Level**							
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**
2	8	7	6	7	7	6	8	6
4	0	4	6	9	7	9	2	0
6	5	5	5	5	2	2	1	1
8	8	3	6	8	5	3	2	6
9	5	8	8	7	7	7	4	5

The input, output, and internal variables of the CBLA Nodes and non-CBLA Nodes during the entire trial were collected. The activation levels of each CBLA Node were extracted at 1.0s time intervals for analyzing the correlations between activations and interest levels. A time interval of 1.0s was chosen to give sufficient granularity to the data. Other similar values could have been chosen and it would not affect the results of the analysis significantly.

5.4.6 Analysis I – Average Interest Levels between Prescripted Mode and CBLA Mode

We first separated the levels of interest data on the questionnaire cards (Tables 5.3 and 5.4) into four groups: Prescripted Mode (first half), Prescripted Mode (second half), CBLA Mode (first half), and CBLA Mode (second half). The "Prescripted Mode" or "CBLA Mode" indicates the version of the behaviours that was running when the samples were recorded, and "(first half)" or "(second half)" indicates whether the samples were taken during the first or the second half of the trial. We then took the average across all trials for each sample point, which resulted in four sets of data with five elements each.

Prescripted Mode (first half):	(4.75, 2.5, 4.5, 6.25, 7.5)
Prescripted Mode (second half):	(7.0, 4.75, 5.0, 6.25, 7.0)
CBLA Mode (first half):	(6.75, 4.5, 1.5, 4.0, 5.75)
CBLA Mode (second half):	(6.0, 3.25, 3.75, 5.625, 3.25)

We then performed Welch's T-Tests for unequal variance on different combinations of these four sets of data as described below. For each test from Test 1 to Test 6, the two-tails P-value must be below 0.1 in order for the null hypothesis (H0) to be rejected in favour of the alternate hypothesis (H1).

1. Prescripted Mode: on during first half vs one during second half

   Data Set 1: Prescripted Mode (first half)

   Data Set 2: Prescripted Mode (second half)

   — H0. Average interest level for prescripted mode is the same regardless if it is on first or second.

   — H1. Average interest level for prescripted mode is different depending on whether it is on first or second.

2. CBLA Mode: on during first half vs one during second half

   Data Set 1: CBLA Mode (first half)

   Data Set 2: CBLA Mode (second half)

   — H0. Average interest level for CBLA mode is the same regardless if it is on first or second.

   — H1. Average interest level for CBLA mode is different depending on whether it is on first or second.

3. Either Mode: on during first half vs one during second half

   Data Set 1: Prescripted Mode (first half) + CBLA Mode (first half)

   Data Set 2: Prescripted Mode (second half) + CBLA Mode (second half)

   — H0. Average interest level is the same regardless if it is on first or second.

   — H1. Average interest level is different depending on whether it is on first or second.

4. CBLA Mode vs. Prescripted Mode: on during first half

   Data Set 1: CBLA Mode (first half)

   Data Set 2: Prescripted Mode (first half)

   — H0. Average interest level for CBLA mode is the same as Prescripted mode when it is on first.

   — H1. Average interest level for CBLA mode is different from Prescripted mode when it is on first.

5. CBLA Mode vs. Prescripted Mode: on during second half

   Data Set 1: CBLA Mode (Second half)

   Data Set 2: Prescripted Mode (second half)

   — H0. Average interest level for CBLA mode is the same as Prescripted mode when it is on second.

   — H1. Average interest level for CBLA mode is different from Prescripted mode when it is on second.

6. CBLA Mode vs. Prescripted Mode: on during either half

   Data Set 1: CBLA Mode (first half) + CBLA Mode (second half)

   Data Set 2: Prescripted Mode (first half) + Prescripted Mode (second half)

   — H0. Average interest level for CBLA mode is the same as Prescripted mode.

   — H1. Average interest level for CBLA mode is different from Prescripted mode.

Out of the six Welch's T-Tests, only Test 5, which compared the levels of interest between CBLA Mode and Prescripted Mode when they were on during the second half of the trial, was able to reject the null hypothesis with a two-tailed P-value of 0.0684. The average

interest level of Prescripted Mode and CBLA Mode were 6.0 and 4.375 respectively. This means that Prescripted Mode was more interesting than CBLA Mode by approximately 37% when they were on during the second half of the trial with a 96.58% confidence. We did not find any significant differences in reported levels of interest between CBLA Mode and Prescripted Mode in any other cases.

Furthermore, we were interested in finding out if an average participant would find Prescripted Mode more or less interesting than CBLA Mode. Since each participant had seen both versions of behaviours, we compared their responses by performing Paired T-Tests.

We first compared each test subject's average levels of interests in CBLA Mode and Prescripted Mode, irrespective of which halves the behaviours were presented.

7. CBLA Mode vs Prescripted Mode

   Data Set 1: CBLA Mode (first half) + CBLA Mode (second half)

   Data Set 2: Prescripted Mode (second half) + Prescripted Mode (first half)

   — H0. Participants find that CBLA Mode is equally as interesting as Prescripted Mode.

   — H1. Participants find that CBLA Mode is not equally as interesting as the Prescripted Mode.

In Test 7, we found that CBLA Mode was in fact less interesting than Prescripted Mode with a 96.16% confidence. An average participant who had interacted with both kinds of behaviours gave Prescripted Mode a rate of 5.55 and a CBLA Mode a rate of 4.44. This means that Prescripted mode was approximately 25% more interesting than CBLA Mode irrespective of the order that they were presented.

Taking a closer look, we wanted to see if the orders that the two versions were presented had any significance.

8. CBLA Mode vs. Prescripted Mode when Prescripted Mode is on first

   Data Set 1: CBLA Mode (second half)

   Data Set 2: Prescripted Mode (first half)

   — H0. Participants find that CBLA Mode is equally as interesting as Prescripted Mode when Prescripted Mode is on first.

— H1. Participants find that CBLA Mode is not equally as interesting as the Prescripted Mode when Prescripted Mode is on first.

9. CBLA Mode vs Prescripted Mode when CBLA Mode is on first

   Data Set 1: CBLA Mode (first half)

   Data Set 2: Prescripted Mode (second half)

   — H0. Participants find that CBLA Mode is equally as interesting as Prescripted Mode when CBLA Mode is on first.

   — H1. Participants find that CBLA Mode is not equally as interesting as the Prescripted Mode when CBLA Mode is on first.

In Test 8, we could not find any significance difference in ratings between CBLA Mode and Prescripted Mode when Prescripted Mode was on first. On the other hand, in Test 9, we found that CBLA Mode was rated lower at an average of 4.5 compared to Prescripted Mode at an average of 6.0 with a 96.32% confidence when CBLA Mode was on first. This translates to a 33% increase in interest level when the behaviour was switched from CBLA Mode to Prescripted Mode. This is quite surprising as we expected the initial curiosity of the participants would boost the interest level of the behaviour presented first.

5.4.7 Analysis II – Correlations between Activation level and Interest Level

This analysis aims to determine if there were any correlations between the participants' reported levels of interest to the activation levels of the sculpture. In this analysis, we did not consider which version of the behaviours was running at the sample point and focused on the actual activation levels of the sculpture.

The extracted activation levels for each CBLA Node were the output levels of each node averaged over 1.0s windows. For each trial, there were 8 sample points which correspond to the times when the participant filled out questionnaire cards and reported his or her levels of interest and locations. For each sample point, the time interval containing it and 30 time intervals (which translates to 30 seconds) preceding it were considered as activations related to the sample point. A time interval of 30s was chosen in order to include prior activations that were likely to be associated with the participant's response at that a sample point.

Two metrics were developed to quantify activation levels. The first metric is "average sample average activation", $\overline{\overline{\alpha}}$. It is computed by taking the average of the output values in the 30 time intervals preceding and at the sample point and then taking the average of that value across all the Nodes under consideration as formulated in (5.5).

$$\overline{\overline{\alpha}} = \frac{1}{N}\sum_{j=0}^{N}\frac{1}{31}\sum_{t=t_{s-30}}^{t_s}\overline{m}_{jt} \tag{5.5}$$

where $\overline{\overline{\alpha}}$ is the average sample average activation level among all CBLA Nodes being considered; N is the number of CBLA Nodes being considered; t_s is the time interval that the sample point is in. t_{s-30} is the time interval that is 30 time intervals before t_s; and $\overline{m}_{jt}$ is the output level of CBLA Node j at time interval t.

The second metric is the "average sample peak activation", $\overline{\widehat{\alpha}}$. It is computed by taking the maximum output values in the 30 time intervals preceding and at the sample point and then taking the average of that value across all the nodes under consideration as formulated in (5.6).

$$\overline{\widehat{\alpha}} = \frac{1}{N}\sum_{j=0}^{N}\max(\overline{m}_{jt_{s-30}},...,\overline{m}_{jt_s}) \tag{5.6}$$

where $\overline{\widehat{\alpha}}$ is the average sample peak activation level among all CBLA Nodes being considered; N is the number of CBLA Nodes being considered; $\overline{m}_{jt_{s-30}}$ and $\overline{m}_{jt_s}$ are the output levels of the CBLA Node j at time intervals t_{s-30} and t_s respectively.

Using those two metrics, we examined if there were any correlations between interest levels and different types of activation levels. One type of activation level is the system wide activation which was captured by computing the metrics among all CBLA Nodes. The other type is device type activation, which was captured by computing the metrics among just Half-Fin Nodes, Light Nodes, or Reflex Nodes.

In addition, proximities of the activations to the participant were considered. We formulated a modified version of $\overline{\overline{\alpha}}$ and $\overline{\widehat{\alpha}}$ that are weighted by the inverse proximities of the Nodes to the participant. They are called "average sample proximal average activation", $\overline{\overline{\rho}}$, and "average sample proximal peak activation", $\overline{\widehat{\rho}}$. The proximity was measured in relative distance unit as only the relative proximities among Nodes were considered in this metric.

$$\overline{\overline{\rho}} = \frac{1}{N}\sum_{j=0}^{N}\frac{1}{31\cdot d_j}\sum_{t=t_{s-30}}^{t_s}\overline{m}_{jt} \tag{5.7}$$

where $\overline{\overline{\rho}}$ is the average sample proximal average activation level among all CBLA Nodes being considered; N is the number of CBLA Nodes being considered; d_j is the relative distance between the CBLA Node j and the subject; t_s is the time interval that the sample point is in; t_{s-30} is the time interval that is 30 time intervals before t_s; and $\overline{m}_{jt}$ is the output level of CBLA Node j at time interval t.

$$\overline{\widehat{\rho}} = \frac{1}{N}\sum_{j=0}^{N}\frac{1}{d_j}\max(\overline{m}_{jt_{s-30}}, ..., \overline{m}_{jt_s}) \tag{5.8}$$

where $\overline{\widehat{\rho}}$ is the average sample proximal peak activation level among all CBLA Nodes being considered; N is the number of CBLA Nodes being considered; d_j is the relative distance between the CBLA Node j and the subject; $\overline{m}_{jt_{s-30}}$ and $\overline{m}_{jt_s}$ are the output levels of the CBLA Node j at time intervals t_{s-30} and t_s respectively.

Since we were interested in the correlations between activation levels of the sculpture and the interest level of an average user, we combined the data for all 10 trials and computed metrics described above. We then computed the Pearson correlation coefficients, r, between each of those metrics and the user's level of interest in Table 5.5.

Table 5.5: Correlations between activation levels and user's interest level

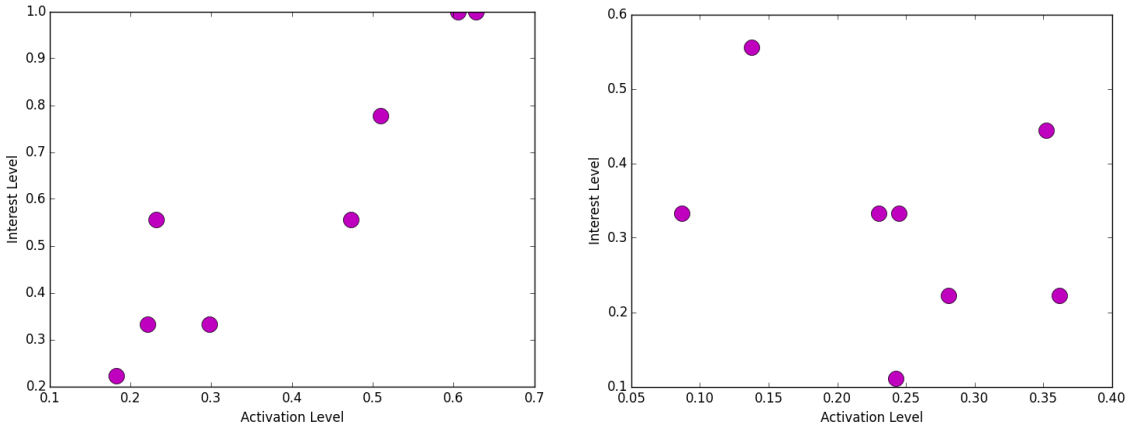
Metric	**Nodes**	r	**P-value**
$\overline{\overline{\alpha}_{all}}$	all CBLA Nodes	0.366304	0.000833
$\overline{\overline{\alpha}_{hf}}$	Half-Fin Nodes	0.357302	0.001139
$\overline{\overline{\alpha}_{l}}$	Light Nodes	0.395201	0.000286
$\overline{\overline{\alpha}_{rfx}}$	Reflex Nodes	-0.083434	0.461854
$\overline{\widehat{\alpha}_{all}}$	all CBLA Nodes	0.317221	0.004143
$\overline{\widehat{\alpha}_{hf}}$	Half-Fin Nodes	0.374608	0.000618
$\overline{\widehat{\alpha}_{l}}$	Light Nodes	0.322380	0.003541
$\overline{\widehat{\alpha}_{rfx}}$	Reflex Nodes	0.021213	0.851844
$\overline{\overline{\rho}_{all}}$	all CBLA Nodes	0.345267	0.001709
$\overline{\widehat{\rho}_{all}}$	all CBLA Nodes	0.317167	0.004149

Over the whole system, there was a weak positive correlation between interest level and both average sample average activation levels, $\overline{\overline{\alpha}_{all}}$, and average sample peak activation levels, $\overline{\widehat{\alpha}_{all}}$, with over 99% confidence.

For device type correlation, we found a higher correlation for Light Nodes than the system-wide correlation when considering the average sample average activation levels. On the other hand, higher correlation was found for Half-Fin Nodes when considering the average sample peak activation levels. Interestingly, we did not find any significant correlation between the activation levels of Reflex Node to the user's interest level.

The proximities of the activations do not increase the correlation. The correlation coefficients for average sample proximal average activation $\overline{\overline{\rho}_{all}}$ and average sample proximal peak activation $\overline{\widehat{\rho}_{all}}$ are approximately the same as their system-wide counterparts. This might be attributed to the fact that the sculpture is relatively small.

Although the overall correlations across all ten studies were relatively weak, there were in fact large variations among different studies. For instance, the average total peak activations level, $\overline{\widehat{\alpha}_{all}}$, for Study 10 showed strong correlation with the subject's interest level. It had a Pearson coefficient, r, of 0.924443 with over 98% confidence. On the other hand, the same metric for Study 3 showed no significant correlation. The two cases are plotted in Figure 5.12 for a visual side-by-side comparison.



(a) Study 10 with strong correlation (b) Study 3 with no significant correlation

Figure 5.12: Study-specific correlation between the average sample peak activation level, $\overline{\widehat{\alpha}_{all}}$, and user's level of interest.

This shows that different people responded and interacted very differently to the behaviours of the sculpture. For instance, some people might indeed be attracted by the activations and were engaged to interact with the sculpture, while some might be attracted to the sculpture due to other reasons, such as the aesthetic of the design. More samples would be needed to further categorize the different types of users.

5.4.8 Analysis III – Responsiveness

In the exit questionnaire, we asked the participants to rate their overall interest level and how responsive they thought the behaviour was. In Table 5.6, responses for the two questions and the average interest level reported on the questionnaire cards were tabulated. All three ratings were reported on a scale of 0 to 9.

Table 5.6: User reported overall interest levels and responsiveness for each trial. The trials are further grouped into Prescripted First and CBLA First.

	Trial #	Q1. Overall Interest Level	Q2. Responsiveness	Average Interest Level on Questionnaire Cards
Prescripted First	1	6	4	5.375
	3	6	5	2.875
	5	6	6	4.125
	7	7	6	5.938
	10	9	7	5.375
CBLA First	2	8	8	6.875
	4	7	4.5	4.625
	6	6	6	3.250
	8	7	7	5.125
	9	8	9	6.375

The Pearson correlation coefficient, r, between responsiveness and overall interest level was 0.680 with over 96% confidence. Similarly, the r between responsiveness and the average interest level reported on the questionnaire cards was 0.572 with over 91% confidence. This shows that responsiveness of the behaviours did have a moderate positive correlation to how interesting they were to the users.

On the other hand, there was moderate discrepancy between the overall interest level and the average interest level reported on the questionnaire cards. Taking the difference between the two revealed that, on average, the overall interest level rating was greater than average interest level reported on the questionnaire cards by 2.01 points. Their Pearson correlation coefficient is 0.682 with over 97% confidence. These results show that

the participants' reported interest levels as they were interacting with the sculpture were significantly different from the ones reported afterwards. This is probably because some people tended to put more emphasis on their most recent experience when answering the questions on the exit questionnaires. In fact, if Prescripted Mode was on during the second half of the trials, the average responsiveness rating was 6.9 as opposed to 5.6 when CBLA Mode was on during the second half with confidence level over 89%. Since the CBLA Mode should be less responsive than Prescripted Mode, we can speculate that when subjects reported on the their impressions on the responsiveness of the sculpture after the trials, they disproportionately emphasized their most recent interactions.

5.4.9 Discussion

In this user study, we attempted to answer the three research questions raised in Section 5.4.1.

First, does the use of the CBLA increase user's interest level over prescripted behaviours? In the analyses done in Section 5.4.6, we showed that CBLA Mode was in fact less interesting than Prescripted Mode by 25%. This effect was even more pronounced when the trial started with CBLA Mode first and switched to Prescripted Mode halfway. Prescripted Mode was on average 33% more interesting that CBLA Mode. This is contrary to our hypothesis as we did not find any significant evidence that the use of CBLA increased the users' interest levels.

Second, do people perceive CBLA as non-random? It is more difficult to answer this question since we did not ask the participants to rate the sculpture's responsiveness for each type of behaviour. In fact, based on question 4 of the exit questionnaire, most participants did not realize that there were two distinct sets of behaviours that were switched over midway. This wasn't asked since we couldn't reveal the type of behaviours that they should expect during the trials. However, we did find a positive correlation between interest level and responsiveness rating. Since CBLA Mode was considered less interesting when we tried to answer research question 1, this might indicate that CBLA Mode was considered less responsive. In addition, we speculate that test subjects would emphasize the later half of the trials when answering the exit questionnaires. On that front, we also found that the responsiveness ratings were lower when CBLA Mode was on in the second half. In addition, the written responses when the participants were asked to described the behaviours of the sculpture, they often described the CBLA portion using words like "random", "totally random", "somewhat random", or "unresponsive". This shows that many participants did think that CBLA Mode was indeed random. This is also contrary to our hypothesis that the participants can perceive the learning behaviours as non-random.

Finally, are certain behaviours more interesting than others? In the analysis done in Section 5.4.7, we found that there was a weak positive correlation between the overall activation levels and the user's interest level. In other word, the participants found that more actuation is more interesting than less actuation. On the other hand, we did not find that any specific device had a significantly stronger positive correlation. However, we found that activations of a Reflex Node, which actuates either a pair of LEDs or a vibration motor, shows no correlation with the user's interest level. Similarly, we did not find that proximity of the actuations influenced the user's level of interest much either. This may be because the size of the sculpture was relatively small and the participants could easily see, hear, and walk to anywhere in the space relatively quickly, in comparison to the time required to fill out the questionnaire cards. In sum, we show that activations do tend to increase user's level of interest but, contrary to our hypothesis, we could not find any particular categories of behaviours that were more interesting than the others.

However, these conclusions may only apply to this particular set-up under this particular set of procedures. For instance, we hypothesize that, over a long period of time, prescripted behaviour would become repetitive and less interesting. In this user study, the participants were only interacting with the prescripted behaviours for 10 minutes and perhaps that was insufficient for them to realize that it was repetitive and lose interest. In fact, one participant thought it was responsive to sound, and was making noise and still trying to figure out its non-existent response to sound until the end.

Moreover, the set up of this interactive art sculpture was also very different from a typical set-up. Typically, this kind of sculpture is set up as an art exhibition in some kinds of public space like department stores, office buildings, and museums. Visitors are free to enter or leave the space as they like. In cases of a permanent installation, the sculpture is placed in the background to some other daily activities. This is very different from the one-on-one, timed interactions that we tested in this user study. In addition, typically, visitors would be accompanied by other people and the effect of multiple occupants in the same space was not covered.

In addition, this project is motivated by the desire to generate life-like behaviours automatically. We don't know if being life-like is interesting to all people. In fact, from the informal conversations after the studies, we realized that people had very different interests, and different expectations about the behaviours of the sculpture. Some expected much more coordinated, fast-pace movements, while some found the slow and organic-looking movements appealing. Some expected the sculpture to be very responsive, while some did not even know that there were sensors that could detect their presence at first. Some participants took figuring out the exact mechanism of sculpture's behaviours as a challenge; some were enjoying it and some were frustrated by their abilities to figure out

the patterns. Moreover, there were also participants who enjoyed looking at the design of the sculpture and some were interested in the design of the circuit boards and actuators. They spent a great deal of time examining the details of the sculpture itself rather than interacting with the sculpture.

Furthermore, the ways how the participants interacted with the sculpture varied greatly. Some mainly stood back and observed, while some walked around the space rapidly and touched many parts of the sculpture at great frequency and perhaps randomly. In fact, one participant was taking apart the sculpture in order to better examine the parts and how those alterations change the activation patterns of the sculpture.

This shows that the data should be further categorized based on the type of user. Different people expected different things out of this experience, and analyzing the different groups separately may reveal more useful information about user's response to the different types of behaviours. In addition, the user study should be done in a more realistic setting over a longer period of time to reveal whether CBLA can indeed be more interesting in the long run.

Chapter 6

Conclusions and Future Work

APPENDICES

Appendix A

User Study Materials

A.1 Information and Consent Form

Information & Consent Form

Date:

Title of Project:
Investigation of user's level of interest while interacting with the Sentient Canopy interactive art sculpture

Faculty Supervisors:

Dr. Dana Kulić
Department of Electrical and Computer Engineering
dana.kulic@uwaterloo.ca
519-888-4567 ext. 37260

Dr. Robert B. Gorbet
Department of Knowledge Integration
rbgorbet@uwaterloo.ca
519-888-4567 ext. 33489

Student Investigator:
Matthew Tsz Kiu Chan
Department of Electrical and Computer Engineering
matthew.chan@uwaterloo.ca

Study Overview

My name is Matthew Tsz Kiu Chan and I am a MASc student working under the supervision of Dr. Dana Kulic in the Adaptive Systems Laboratory in the Department of Electrical and Computer Engineering and Dr. Rob Gorbet in the Department of Knowledge Integration, at the University of Waterloo.

You are invited to participate in a study that investigates users' level of interest while interacting with the new series of interactive art sculptures named *Sentient Canopy*, which is produced in collaboration with the Philip Beesley Architect Inc. (PBAI) in Toronto, ON. This study is for my Master's thesis and is conducted in collaboration with PBAI. Similar sculptures have been displayed in many venues in different countries in the past. For a list of previous installations, please visit http://philipbeesleyarchitect.com/sculptures.

What You Will Be Asked to Do

Participation in this study involves interacting with the art sculpture. While you are interacting with the sculpture, you will be asked to fill out a short questionnaire regarding your interest level whenever you hear a signal. You are free to roam around the studio and interact with the sculpture as you like. There is not any specific prescribed interaction and you are free to choose how you want to interact with the sculpture. At the end of the study, you will return all the completed questionnaires in an envelope and you will be asked to fill out

another short questionnaire about your overall experience. Please see attached "Study Procedures" sheet for details.

Participation

Participation in this study is voluntary and it will take approximately 30 minutes of your time. You may decline to answer any questions presented during the study if you so wish. Further, you may decide to withdraw from this study at any time by advising the researcher, and may do so without any penalty.

Personal Benefits of the Study

There are no personal benefits to participation.

Risks to Participation in the Study

There are no known or anticipated risks/stressors to the participants as a result of taking part in this study. The interactive art sculptures developed by Philip Beesley Architect Inc. have been exhibited in more than 10 countries where thousands of people have visited the art sculptures. The sculptures are made of soft and light-weight material and actuated with low power actuators. An actuator is a device that converts energy, such as electricity, into motion, light, and sound. This new series of interactive art sculptures does not pose greater risk of physical harm than what existed in previous installations.

Confidentiality

All information you provide is considered completely confidential; indeed, your name will not be included or in any other way associated, with the data collected in the study. Furthermore, because the interest of this study is in the average responses of the entire group of participants, you will not be identified individually in any way in any written reports of this research. Paper records of data collected during this study will be destroyed after they have been digitized. The converted electronic data will be kept for 20 years on a secure computer, to which only researchers associated with the Adaptive Systems Laboratory have access. This data may be additionally used for subsequent secondary data analysis comparing user responses to various interaction strategies, however, your name and identity will not be obtainable from the stored data.

Questions and Research Ethics Clearance

If after receiving this letter, you have any questions about this study, or would like additional information to assist you in reaching a decision about participation, please feel free to ask the student investigator or a faculty supervisor listed at the top of this sheet.

I would like to assure you that this study has been reviewed and received ethics clearance through a University of Waterloo Research Ethics Committee. However, the final decision about participation is yours. If you have any comments or concerns resulting from your participation in this study, please contact Dr. Maureen Nummelin, the Director, Office of Research Ethics, at 1-519-888-4567, Ext. 36005 or maureen.nummelin@uwaterloo.ca.

Thank you for your interest in our research and for your assistance with this project.

Consent of Participant

I have read the information presented in the information letter about the study being conducted by Matthew Tsz Kiu Chan under the supervision of Dr. Dana Kulic in the Adaptive Systems Laboratory in the Department of Electrical and Computer Engineering and Dr. Rob Gorbet in the Department of Knowledge Integration at the University of Waterloo in collaboration with Philip Beesley Architect Inc. in Toronto, ON.

I have had the opportunity to ask any questions related to this study, to receive satisfactory answers to my questions, and any additional details I wanted. I am aware that I may withdraw from the study without any penalty at any time by advising the researchers of this decision.

This project has been reviewed by, and received ethics clearance through a University of Waterloo Research Ethics Committee. I was informed that if I have any comments or concerns resulting from my participation in this study, I may contact the Director, Office of Research Ethics, at 1-519-888-4567, Ext. 36005.

With full knowledge of all foregoing, I agree, of my own free will, to participate in this study.

_____

Print Name

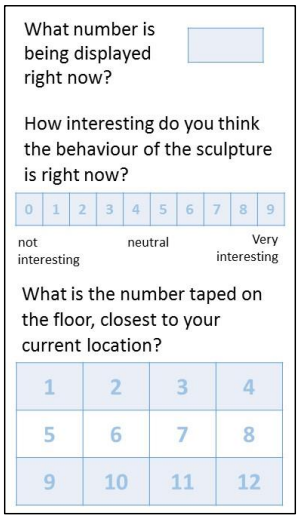
_____

Signature of Participant

_____

Date

Study Procedures



1. In the envelope that you have received, you will find a stack of questionnaire cards that look like the image to the right.
2. You are free to walk around the studio and interact with the art sculpture. Periodically, a tone will sound in the studio. When you hear the tone, please take one card out. You are expected to hear a sound around every 3 to 5 minutes.
3. When the tone sounds, the researcher will display a number; and write down the number in the first box.
4. Report how interesting you think the behaviour of the sculpture is at that moment by circling one of the numbers from 0 to 9 under the second question.
5. Look down to the floor. There should be a number right below or near where you are standing. Please circle the number that is closest to where you are standing under the third question.
6. After that, you can return the completed card to the envelope and continue walking around the studio and interacting with the sculpture.
7. When you leave or when the staff signal the end of the showing, please return the envelope containing the questionnaire cards to the desk where you got the envelope from.
8. You will then be asked to fill out a survey on your overall experience and your knowledge of machine learning.
9. At last, you will be given a debriefing letter with more information about the study.

A.2 Exit Questionnaire

Feedback Survey

1. How interesting was your overall experience while interacting with the art sculpture?

0	1	2	3	4	5	6	7	8	9

Not Interesting — Neutral — Very Interesting

2. How responsive was the sculpture to your presence? Do you think its behaviour was totally random, or do you think it was responding directly to you?

0	1	2	3	4	5	6	7	8	9

Totally Random — Unambiguous Response

3. How familiar are you with machine learning algorithms?

0	1	2	3	4
I've never heard of it	I've heard of the term but don't know what it is	I've a general idea of how it works	I've studied or done some work in machine learning	I'm an expert in machine learning

4. How would you describe the behaviour of the sculpture?

5. Additional comments

A.3 Debriefing Letter

DEBRIEFING LETTER

Adaptive System Laboratory

Department of Electrical and Computer Engineering, University of Waterloo

Project Title:
Investigation of user's level of interest while interacting with the Sentient Canopy interactive art sculpture

Student Investigator:
Matthew Tsz Kiu Chan
Department of Electrical and Computer Engineering
matthew.chan@uwaterloo.ca

Faculty Advisors:
Dr. Dana Kulić
Department of Electrical and Computer Engineering
dana.kulic@uwaterloo.ca
519-888-4567 ext. 37260

Dr. Robert B. Gorbet
Department of Knowledge Integration
rbgorbet@uwaterloo.ca
519-888-4567 ext. 33489

We appreciate your participation in our study, and thank you for spending the time helping us with our research!

In this study, you interacted with the *Sentient Canopy* interactive art sculpture and were asked to provide feedback on your interest level as you interacted with the sculpture, and on your overall experience after the session.

Two interactive behaviours were actually run on the sculpture during your visit. One was a pre-programmed version in which the automatic actuation and responses to sensory inputs were fixed. The other set of behaviours was based on a Curiosity-Based Learning Algorithm (CBLA), in which parameters that dictate the behaviours are generated automatically and change as the sculpture learns about itself and you. In the CBLA version, the sculpture will continuously learn to model the mapping between its inputs and outputs and select actions that will lead to greater learning. For example, it will attempt to learn if moving the Tentacle up and down will affect readings from the sensors. This is analogous to how animals and human beings learn. In this study, we hope to determine whether the behaviours generated through this method can make the experience of interacting with the sculptures more interesting. Simply put, we would like to know if behaviours generated using the CBLA are more interesting than those designed by human experts. The results of this study can enable designers to design more engaging and interesting art sculptures.

We hypothesized that the viewers will enjoy behaviours that are less predictable and changing. It is expected that since the CBLA continuously generates new interactive behaviours, viewers will find it more engaging and interesting. On the other hand, although the CBLA continuously generates new

behaviours, it is not random. It tends to exhibit new behaviours that promote predictable response from the viewers and the environment. We think that the viewer will be able to recognize that its behaviours are not random. In addition, we hypothesized that certain sets of parameters may generate more interesting behaviours and this can be a systematic way to discover those behaviours. The results from this study will provide us with evidence allowing us to accept or reject these hypotheses.

We could not give you complete information about the study before your involvement because it may have influenced your perception during the study in a way that would make investigations of the research question invalid. Specifically, we did not tell you that there were two versions of the behaviours and what the expected behaviours of the sculpture are because we wanted to avoid the *subject-expectancy effect*. This is a form of bias in which the test subject expects a particular result and this unconsciously affects the outcome. We hope that you understand the need for this partial disclosure now that the purpose of the study has been more fully explained to you.

All information you provided is considered completely confidential; indeed, your name will not be included or in any other way associated, with the data collected in the study. Furthermore, because the interest of this study is in the average responses of the entire group of participants, you will not be identified individually in any way in any written reports of this research. Paper records of data collected during this study will be destroyed after they have been digitalized. The converted electronic data will be kept for 20 years on a secure computer, to which only researchers associated with the Adaptive Systems Laboratory have access. This data may be additionally used for subsequent secondary data analysis comparing user responses to various interaction strategies, however, your name and identity will not be obtainable from the stored data.

This project has been reviewed by, and received ethics clearance through a University of Waterloo Research Ethics Committee. In the event you have any comments or concerns resulting from your participation in this study, please contact Dr. Maureen Nummelin, the Director, Office of Research Ethics, at 1-519-888-4567, Ext. 36005 or maureen.nummelin@uwaterloo.ca.

Because the study involves some aspects that you were not told about before starting, it is very important that you not discuss your experiences with anyone who potentially could be in this study. If people come into the study knowing about our specific predictions, as you can imagine, it could influence the results, and the data we collect would be not be useable. Also, since you will be given a copy of this feedback letter to take home with you, please do not make this available to others.

If you think of some other questions regarding this study, please do not hesitate to contact Matthew T.K. Chan at matthew.chan@uwaterloo.ca.

We really appreciate your participation, and hope that this has been an interesting experience for you.

References (related studies that may be of interest to you):

P.-Y. Oudeyer, F. Kaplan and V. V. Hafner, "Intrinsic Motivation Systems for Autonomous Mental Development," *IEEE Transactions on Evoluntionary Computation,* vol. 11, no. 2, pp. 265-286, 2007.

References

[1] M. T. K. Chan, R. Gorbet, P. Beesley, and D. Kulić, "Curiosity-Based Learning Algorithm for Distributed Interactive Sculptural Systems," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, (Hamburg Germany), pp. 3435–3441, 2015.

[2] R. Gorbet, M. Memarian, M. Chan, D. Kulic, and P. Beesley, "Evolving Systems within Immersive Architectural Environments: New Research by the Living Architecture Systems Group," *Next Generation Building*, 2015.

[3] P. Beesley, R. Gorbet, P. Ohrstedt, and H. Isaacs, *Hylozoic Ground: Liminal Responsive Architecture*. Toronto, Ontario, Canada: Riverside Architectural Press, 2010.

[4] P. Beesley and C. Macy, *Kinetic Architectures & Geotextile Installations*. Toronto, Ontario, Canada: Riverside Architectural Press, 2010.

[5] P. Beesley, *Sibyl: New Responsive Immersive Architecture Projects 2010-12*. Kitchener, Ontario: Riverside Architectural Press, 1 ed., 2012.

[6] P.-Y. Oudeyer, F. Kaplan, and V. V. Hafner, "Intrinsic Motivation Systems for Autonomous Mental Development," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 2, pp. 265–286, 2007.

[7] K. J. Cios and M. E. Shields, "The handbook of brain theory and neural networks," in *Neurocomputing*, vol. 16, pp. 259–261, MIT Press, 1997.

[8] B. Bridgeman, "Efference copy and its limitations," *Computers in Biology and Medicine*, vol. 37, no. 7, pp. 924–929, 2007.

[9] E. Edmonds, G. Turner, and L. Candy, "Approaches to interactive art systems," in *Proceedings of the 2nd international conference on Computer graphics and interactive techniques in Austalasia and Southe East Asia GRAPHITE 04*, pp. 113–117, 2004.

[10] J. Drummond, "Understanding Interactive Systems," *Organised Sound*, vol. 14, no. 02, p. 124, 2009.

[11] B. Costello, L. Muller, S. Amitani, and E. Edmonds, "Understanding the experience of interactive art: Iamascope in Beta_space," *IE '05 Proceedings of the second Australasian conference on Interactive entertainment*, pp. 49–56, 2005.

[12] B. Wands, "Variations : An Interactive Musical Sculpture," in *C&C '05 Proceedings of the 5th conference on Creativity & cognition*, pp. 306–309, ACM, 2005.

[13] A. D. Dragan, S. Bauman, J. Forlizzi, and S. S. Srinivasa, "Effects of Robot Motion on Human-Robot Collaboration," in *Proceedings of the Tenth Annual ACM/IEEE International Conference on Human-Robot Interaction - HRI '15*, vol. 1, pp. 51–58, 2015.

[14] Anca Draga, R. Holladay, and S. Srinivasa, "An Analysis of Deceptive Robot Motion," in *Robotics: Science and Systems*, p. 10, 2014.

[15] M. J. Gielniak, K. Liu, and A. L. Thomaz, "Generating human-like motion for robots," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1275–1301, 2013.

[16] M. A. Bedau, J. S. McCaskill, N. H. Packard, S. Rasmussen, C. Adami, D. G. Green, T. Ikegami, K. Kaneko, and T. S. Ray, "Open problems in artificial life," *Artificial life*, vol. 6, no. 4, pp. 363–376, 2000.

[17] Minimaforms, "Petting Zoo FRAC Centre."

[18] T. Ikegami, "A design for living technology: Experiments with the mind time machine," *Artificial Life*, vol. 19, no. 3/4, pp. 387–400, 2013.

[19] M. Lungarella, G. Metta, R. Pfeifer, and G. Sandini, "Developmental robotics: a survey," *Connection Science*, vol. 15, no. 4, pp. 151–190, 2003.

[20] M. Asada, K. Hosoda, Y. Kuniyoshi, H. Ishiguro, T. Inui, Y. Yoshikawa, M. Ogino, and C. Yoshida, "Cognitive Developmental Robotics : A Survey," *IEEE Transactions on Autonomous Mental Development*, vol. 1, no. 1, pp. 12 – 34, 2009.

[21] V. R. Kompella, M. F. Stollenga, M. D. Luciw, and J. Schmidhuber, "Explore to see, learn to perceive, get the actions for free: SKILLABILITY," in *2014 International Joint Conference on Neural Networks (IJCNN)*, pp. 2705–2712, 2014.

[22] P.-Y. Oudeyer, F. Kaplan, V. V. Hafner, and A. Whyte, "The playground experiment: Task-independent development of a curious robot," in *In Proceedings of the AAAI Spring Symposium on Developmental Robotics*, (Stanford, California), pp. 42–47, 2005.

[23] M. Tokic, "Adaptive ϵ-greedy exploration in reinforcement learning based on value differences," in *33rd Annual German Conference on AI*, (Karlsruhe, Germany), pp. 203–210, 2010.

[24] R. Tibshirani, "Regression Shrinkage and Selection via the Lasso," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 58, no. 1, pp. 267–288, 1996.

[25] D. Beazley, "Understanding the Python GIL," in *PyCON Python Conference 2010*, (Atlanta, Gerogia, USA), 2010.

[26] Encyclopædia Britannica, "Joule's law," 2016.

[27] Dynalloy Inc., "Technical Characteristics of Flexinol Actuator Wires (Rev. 12)."

[28] L. C. Burmeister, *Convective Heat Transfer*. John Wiley & Sons, 1993.

[29] Cree Inc., "Cree 5mm Round LED," 2011.