Implementation of attentional bistability of the dragonfly visual neurons in an intelligent biomimetic agent
— Final Report —

Juan Carlos Farah, Panagiotis Almpouras, Ioannis Kasidakis, Erik Grabljevec, Christos Kaplanis
{jcf214, pa512, ik311, eg1114, ck2714}@doc.ic.ac.uk

Supervisors: Professor Murray Shanahan, Zafeirios Fountas, Pedro Mediano
Course: CO530/533, Imperial College London

13^{th} May, 2015

Contents

1 Introduction

Dragonflies are insects of the order Odonata, suborder Anisoptera [6]. They are notoriously effective at prey capture, with success rates ranging from 76% to 97% [7], making the neural processes that underlie this ability particularly interesting to investigate. There has been substantial research into the visual system of dragonflies [8,9] but what seems to be lacking is an effective tool that links models of the various mechanisms involved in this process together, which could help us better understand the particular function of each layer within the system.

At the core of this apparatus, the centrifugal small target motion detector one neuron (CSTMD1) is a higher order visual neuron in the brain of the dragonfly that reacts to the presentation of multiple visual stimuli by firing as if only one of the stimuli was present. This is presumably an attentional selection mechanism [9]. At Professor Murray Shanahan's lab, researchers have simulated the large contralateral dendritic field of the CSTMD1 neuron with a biophysical multi-compartmental Hodgkin-Huxley model. Along with Klaus Stiefel [2], they found that with certain numbers of inhibitory synapses and potassium conductance densities, two mutually-coupled CSTMD1 neurons are capable of a bistable switching process between two input patterns. In order to confirm that this neuron is indeed responsible for target selection, it would be useful to be able to model the CSTMD1 as a part of a whole visual system. Our goal was to create a tool that could hopefully serve the following purposes:

1. Provide a connected model of dragonfly target selection, starting from the visual input to the retina and ending with the motion of the dragonfly.
2. Provide a graphical interface for each module and for the system as a whole, allowing users to run simulations and view useful information on the processing done by each layer.
3. Provide persistent, easily accessible storage of experimental data generated by the individual layers and the system as a whole.
4. Provide a platform for potentially replicating the behaviour of a real dragonfly during prey capture.

1.1 Report Structure

The parts that this report includes are the following:

Specification: This section outlines our original goals and how they were adjusted according to the challenges that were encountered during development. We also present the motivation behind additional goals that were set to ensure a balanced workload and to achieve the best possible results given the high level of complexity and the time constraints of the project.

Design: This section provides an overview of the design for our project. Particular attention is given to the options that were considered and the justification of the choices made for each component individually and for the system as a whole.

Methodology: This section highlights the specific techniques, frameworks and solutions used to meet the goals set forth in the specification. We present the problems and challenges that were faced during the project and explain how we tackled them, with the architecture and implementation of each module being considered in detail. Additionally, we explain our software development strategy and how it helped us address those issues effectively.

Group Work: This section illustrates how we divided our project into small, measurable tasks. We also describe how we divided into sub-groups that could develop each module in parallel and how this allowed us to maximise our throughput given the time constraints.

Final Product: This section present the final product of the work conducted throughout the project, expanding on the goals that were met as well as the goals that were infeasible. We analyse results in detail and motivate future development as well as potential extensions that might be of interest to future collaborators.

Appendix: This section presents a log of the summary of the minutes for meetings conducted throughout the project, covering tasks assigned to each member and his overall contribution.

2 Specification

Our initial approach to establishing the requirements for this project was to brainstorm using goal-oriented capture. As depicted in Figure 1, our initial goal was to create a biomimetic agent that emulates target selection in the dragonfly.

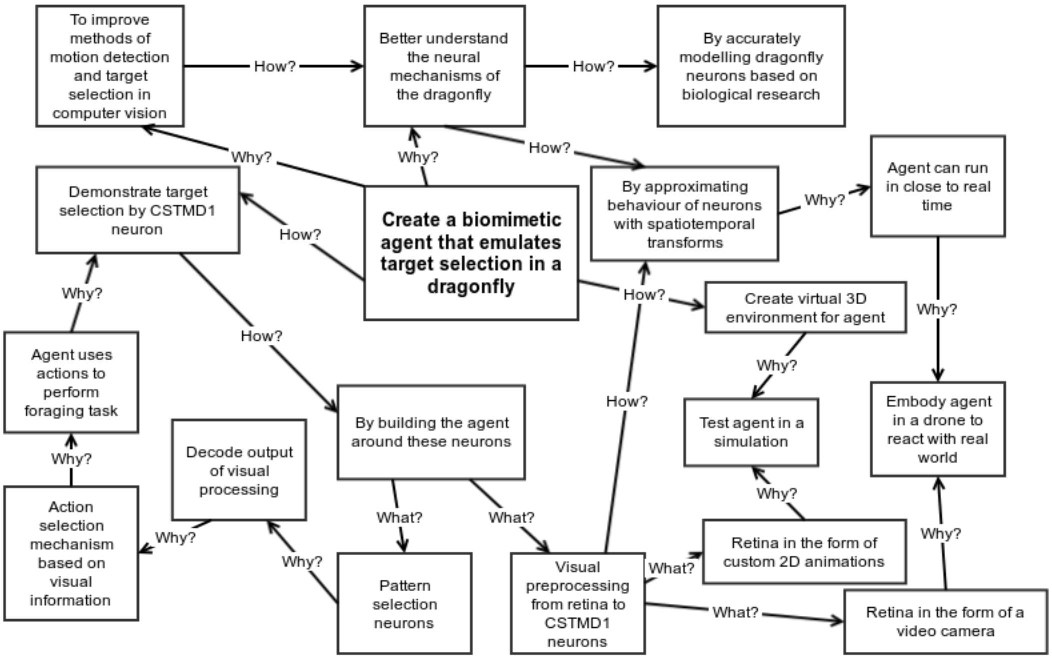

Figure 1: Goal-Oriented Capture Diagram

Goal-oriented capture enabled us to identify the minimum requirements for the completion of our project. As we progressed in the development of the system, we ran into some complications that motivated us to modify the certain aspects of our specification.

2.1 Challenges and Motivation for Updated Specifications

In this section, there is a brief discussion of the challenges that were faced during the project and how they motivated the adjustment of the original specifications.

At the start of the project, our main objective was to create a working dragonfly target selection system. Key to the implementation of a biomimetic agent was the functioning of the CSTMD1 neuron, which had been previously modelled at Professor Shanahan's lab. Having previously displayed evidence of bistability, it was expected that the model of the CSTMD1 would be capable of selecting one target from many in the dragonfly's visual field by replicating the results of research done into real CSTMD1 neurons [9]. Despite the hard work that was put into addressing the issue, the complex morphology of the CSTMD1 neuron made the task infeasible to be fully completed within the time frame of this project. Thus, it was moved to possible extensions and although significant progress was observed, it was only partially completed by the submission deadline.

Furthermore, our difficulties led us to believe that it would be useful for us and for future researchers into dragonfly vision to create a user-friendly interface for interacting with our dragonfly system. To this end, we decided to add the creation of a web client to our requirements. The aim is for the web client to provide an interface that allows the user to run and save simulations of either each module individually or of the whole system and to provide key metrics that demonstrate the functionality of each part.

2.2 Final Specifications

The following table summarises the finalised requirements and the level of completion for each part.

Minimum Requirements (Stage 1)	**Completion**
(Ai) Create an animation tool to generate inputs for visual processing.	Full
(Aii) Build a model for the ESTMD neuron present between the retina and the actual CSTMD1 neurons of a real dragonfly.	Full
(Aiii) Design connection between ESTMD and CSTMD1 neurons.	Full
(B) Build a layer of pattern recognition neurons that can learn to recognise spike patterns within a noisy background.	Full
(C) Integrate the visual processing and pattern recognition system to detect patterns within the CSTMD1 output and add a simple action selection mechanism.	Full

Expected Implementation (Stage 2)	**Completion**
(A) Develop a web client to analyse metrics of each component in our model.	Full
(B) Create an animation for the dragonfly agent.	Full
(C) Enhance the action selection mechanism to control the agent within the environment.	Full

Possible Extensions (Stage 3)	**Completion**
(A) Improve the usability and features of the web client.	Full
(B) Achieve CSTMD1 target selection through experimentation with various parameters and connections with the ESTMD neurons.	Partial
(C) Implement the agent in a quadcopter drone.	None

3 Design

The challenge of designing a tool that could satisfy the overall functionality outlined in the introduction could be distilled down to two main facets:

1. Deciding on the components of the dragonfly visual system and how they would be connected together.

2. Deciding on the user interface and specifying an API for the system.

3.1 Components

Having consulted academic research on the subject of dragonfly vision and discussed with our supervisors, we managed to identify five key components we would need to model in order to implement a coherent system.

1. Target Animation: An animated video that provides the visual input to the system, emulating the retina of the dragonfly. User should be able to specify movements of multiple targets within the visual field and choose the background.

2. Elementary Small Target Motion Detector (ESTMD): The ESTMD neurons are the first layer of visual processing in the dragonfly. They have the general function of identifying and isolating small moving targets, even against a cluttered, moving background (WIEDERMAN 2008). They take arrays of pixel values as input from the animation and output firing rates of neurons to be processed by the next layer.

3. Centrifugal Small Target Motion Detector (CSTMD): The CSTMD neuron is a higher order visual neuron in the brain of the dragonfly. This neuron reacts to the presentation of multiple visual stimuli by firing as if only one of the stimuli was present [9]. The CSTMD takes the firing rates of the ESTMD as input and outputs a time series of neuron spikes (spike trains) to the next layer.

4. Pattern Recognition: This module has the function of detecting patterns in the output of the CSTMD in order to distil features of target movement within the visual field. While it does not have a direct correspondence to a layer of neurons in the dragonfly, it uses a well-established biological learning mechanism called Spike Timing Dependent Plasticity (STDP) [4] [3]. It outputs spike trains to the Action Selection module.

5. Action Selection: This model converts the output of the pattern recognition neurons into movement of the dragonfly, emulating the connection between the visual processing to the motor neurons. It can be trained using STDP combined with reward modulation so that the dragonfly learns to maximise target capture based on the patterns in its visual field. The final output is the original animation with the position of the dragonfly superimposed onto each frame for observation of how effectively it chases targets.

3.2 User Interface

For the user interface we decided that a web client would be the most suitable solution. The web client is designed to be a simple interface that allows for simulations for each of the modules to be run and automated, either separately or jointly. This graphical user interface interacts with each module's API and provides the functionality needed to save and access the results of every experiment. It provides key metrics that can be crucial in understanding the performance of the system. Given the computationally expensive nature of our modules' simulations, a persistent store had to be used to save the output and results of each run. As shown in the structure diagram below, the five modules that our project comprises, are connected sequentially. Saving the output of each module enabled us to reuse the output of a module as input to the next, multiple times without the need to rerun any of the previous modules. What is more, the persistent store allows the web client to provide a view of the results for analysis simply by querying the database instead of regenerating them each time. For our data store MongoDB was chosen. MongoDB provides a fast, scalable solution that does not require strict design decisions in advance. Considering in retrospect the changes and adaptations that had to take place during this project, we trust that MongoDB was a wise choice that contributed to the overall success of this project.

3.3 Design diagram

The following figure provides a graphical representation of the system as whole.

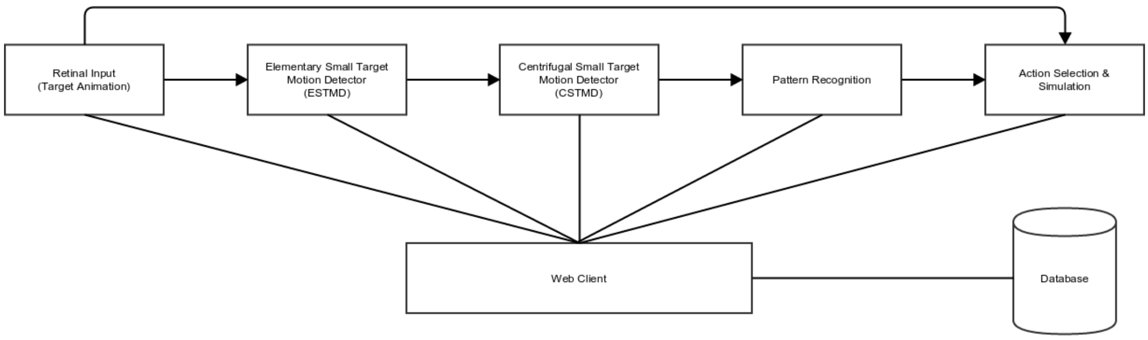

Figure 2: Project Structure Diagram.

3.4 Programming language

We decided that it made sense that the various modules were all programmed in the same language so that they could be easily linked into the web client. The programming language used throughout the project was Python [1]. Python includes libraries that allow for efficient matrix manipulations that were key in most of the modules. Matlab was an alternative we initially considered [5]. Although Matlab is very powerful and user friendly for some particular tasks, it does not allow for the flexibility that Python does. For that reason Matlab was disregarded as an option. The database we used was MongoDB because of the increasing popularity of the distributed databases as well as the fact that this was the database scheme we were most familiar with.

4 Methodology

This section of the report includes a detailed analysis of the methods used to complete the tasks described in the specification section. There was a clear way of dividing the overall project into sub problems as most of the tasks were to a great extent modular and upon completion they required to get connected to each other so as to function as a single system when required. There is also an overview of the testing methodology used throughout this project as well as the coverage achieved by our current regression suite.

For the ease of the reader we enumerate the modules that we developed:

1. Target Animation
2. ESTMD neuron
3. CSTMD neuron
4. Pattern Recognition
5. Action Selection
6. Web Client

4.1 Target Animation

This module represents the input to the dragonfly's retina. We needed to create an animation tool that allows the user to flexibly create a video of moving targets against a custom and potentially moving background. We started with the suggestion of creating a 3D animation where the dragonfly's view would change dynamically depending on the dragonfly's position and orientation but we decided that creating such an animation would be an overkill for our needs and too computationally expensive. Instead we settled with a 2D animation with coloured circles representing the targets against a moving or stationary 2D background. Our priorities were flexibility and simplicity. In the final output of the visual system, we would also want to be able to see the dragonfly chasing targets and we decided that, rather than create a whole new animation where the whole dragonfly is visible, it would make more sense to superimpose the dragonfly's visual focal point as a circle onto the original animation, so that the 'chasing' is represented as what point in its visual field the dragonfly is focusing on.

The design of this module was approached from the user's perspective. The user would want to add targets, a background and get a final video. This could be done with 3 functions that would interact with an interface class. We named that class Animation. There were two types of targets: randomly moving ones and straight-line moving ones. For randomly moving ones, one could set a starting position and speed. In addition one should set a velocity vector for targets moving in straight line. The background is represented with the file location of an image and a speed with which we want it to move. Target and Background are both encapsulated in a class.

The next challenge was to turn this information into an animation. We decided to first create a sequence of images and turn them into an animation at the very end. This would allow us to easily superimpose the dragonfly focal point to the animation in the final stage of the system. We decided upon a Python module called Gizzet (CITATION) for image creation as it offers exactly what we needed drawing circles, adding background images etc. in a user friendly way. We take the user's input to calculate position of each target and of background at each frame. Using that calculation we know where to draw circles and background picture, afterwards we save the created image.

The generated sequence of images is than combined into a video using OpenCV (CITATION) and exported under a selected filename. Additionally, the module outputs the series of images in matrix format of pixel values to be passed to the ESTMD module.

In addition to the above functionality of target animation module, we needed to somehow pass data of target positions to the final module the action selection in order to assist with the reward-modulated learning and to superimpose the position of the dragonfly's focal point onto the animation, which is determined by this final module. To satisfy this need we added functions that calculate positions of targets at a certain time during the animation.

Example of frame in produced video can be seen on the picture 3.

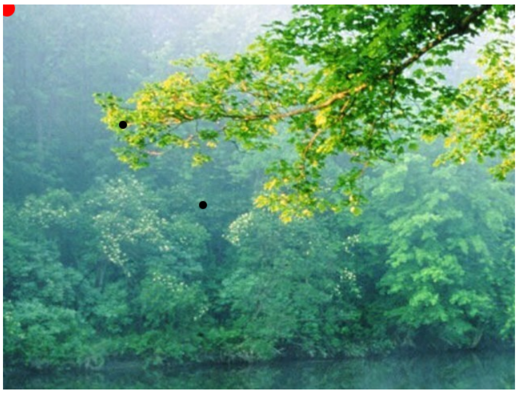

Figure 3: Simple example of action selection output. The red dot represents the dragonfly focal point and the black dot represents the target.

4.2 Elementary Small Target Motion Detector (ESTMD)

In our initial specification, one of our tasks was to connect the retina of our dragonfly to the CSTMD neurons. Having researched how this occurs in real dragonflies, we discovered that we would have to include a layer of neurons that preprocesses the input from the retina before passing it on to the CSTMDs, named elementary small target motion detectors (ESTMD) (WIEDERMAN 2008). While one of the purposes of the CSTMD seems to be to give the ability to select one target among many in the visual field (WIEDERMAN 2013), the function of the ESTMDs seems to be that of identifying small moving targets, even against a cluttered, moving background (WIEDERMAN 2008).

Our research revealed that the ESTMDs actually consist of several layers of neurons required to perform their function, namely photoreceptors, large monopolar cells (LMCs) and rectifying transient cells (RTCs) (WIED 2008). It was immediately obvious to us that implementing all these stages in a multi-compartmental model (the method used for the CSTMD) would be extremely complicated and computationally expensive. Our initial thought was to attempt to use simplified point neuron models (such as Integrate-and-fire) to achieve the ESTMD function, but with a little more research we discovered that it would be possible to do it in a simpler and more computationally efficient way using a series of spatial and temporal transforms on the retina input (WIEDERMAN 2009, HALUPKA 2013). It is important to note that these transforms are intended to directly approximate the layers of biological neurons mentioned above - there are other techniques from computer vision that could be used for small target motion detection, but our priority is to represent the true function of the dragonfly's brain within realistic bounds of computational complexity. The overview of the model we implemented is summarised in Figure 1 and the details of the transforms were taken from (HALUPKA 2013).

The first consideration we had to take was the input and output of the ESTMD module. The input would be a time series of two-dimensional arrays of pixel values from the retina and the output would need to be a time series of neuron spikes (spike trains) to provide input to the CSTMD. The spatial and temporal transforms we intended to use are directly applicable to an array of pixel values but the output of these transforms would also be an array of pixel values that would need to be transformed into spike trains. We decided the simplest and most efficient way of doing this would be to map each pixel value to a corresponding firing rate of a neuron, but this would be dealt with by the CSTMD module. In addition, to demonstrate the effect of the ESTMD module, we decided that it would be useful to provide a video of the output in order to ascertain visually that it is performing its expected function.

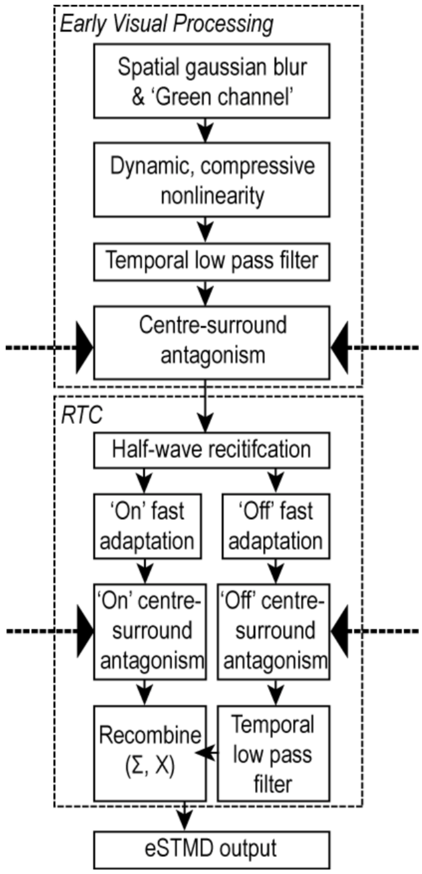

Figure 4: CITE WIEDERMAN 2009

One significant challenge was understanding the transforms and implementing them in Python. For example, understanding the transforms required some background reading on z-transforms, which are the discrete equivalent of Laplace transforms, which are necessary as our system works in discrete time steps. For the translation of the model into Python, we used the Scipy.signal and openCV (CITATION REQUIRED?) libraries extensively: Scipy was used to process the z-transforms (the temporal transformations) and openCV was used for the spatial transforms, such as Gaussian filtering. We also used openCV to convert videos into arrays of pixel values and vice versa.

Once we had programmed the transforms, some tinkering with parameters was required to achieve the small target motion detection we were expecting. In any case, the user of the web client will have the capability of changing these parameters for their own model.

4.3 Centrifugal Small Target Motion Detector Neuron (CSTMD)

The function of this module is to implement a multi-compartmental model of the actual CSTMD neuron that exists in dragonflies, with the hope that it will create an effect of selecting one target from many in the visual field. It takes input from the ESTMD and gives output to the pattern recognition module. The initial code was given to us by our supervisors, who had already shown that there is mutual inhibition of CSTMD1 neurons that could indicate target selection, but it was our job to connect it to the rest of the system and also, as a secondary goal, see if we could replicate the target selection effect observed in real dragonflies. The main issues in tackling this model were:

1. Understanding he complicated nature of the morphologically-modelled CSTMD1
2. Learning how to use the simulation environment of the CSTMD1 neurons (NEURON simulation environment) *CITATION*

3. Efficiently connecting this module with both the ESTMD and the pattern recognition module

4. Generating key indicators to measure the performance and the plausibility of the module, in order to assist the user in assessing the function of this neuron

Given that the CSTMD1 is multi-compartmental model with several parameters that directly or indirectly affect its activity and effect to a given input, we needed to first fully understand the given model before proceeding with any further decisions. Thus, we performed several tests in order be aware of all the plausible values that should be tried for each of the parameters.

Moreover, we had to consider what would be the best way to provide stimuli to the CSTMD1 neurons as an input from the ESTMD module. The NEURON simulation environment provides many classes as a connection to a model from an external input. After examining all the different possibilities we decided to use a leaky integrator (IntFire2() point process) *EXPLAIN MORE ABOUT INTFIRE2???*. More specifically, the ESTMD module provides a time series of neuron spikes which is essentially an array of values each of which corresponds to a spiking rate per retina pixel. Thus, we initiated one leaky integrator for each pixel and provided the spiking rate as its total current. We then connected these point processes to each of the CSTMD neurons used for the simulation. In order to make our model as much biologically plausible as possible, we added some randomized delays to the connections of the leaky integrators with the CSTMD1 neurons with the aim of ensuring that the desired inhibitory effect will be achieved while neurons are provided with some stimuli. We also applied a spatial Gaussian distribution to the weights of these connections, increasing the relative weights of the pixels in the centre of the visual field compared to those on the edges, as this is the case with real dragonflies (*CITE PAPER WIEDERMAN)*.

With regards to the output, since the pattern recognition module requires a considerable number of spike trains as an input, we had to find a way to connect the 2 modules without severely affecting the speed and performance of the CSTMD1 module. Hence, given the morphologically large axon of the CSTMD1 neuron (GEURTEN 2007) we decided that, instead of using as many neurons as the inputs required by the pattern recognition module, to apply a large number of electrodes to different compartments of the neurons and thus provide the adequate number of spike trains without limiting the biological plausibility of our model.

After successfully connecting the different modules, our next goal was to define key performance indicators for the CSTMD1 module in order to measure as accurately as possible its performance and also be able to modify the several parameters of the given model and observe their effect to the simulation. For this purpose, we created several plots which recorded the activity of different compartments of the neurons as well as their spiking rates.

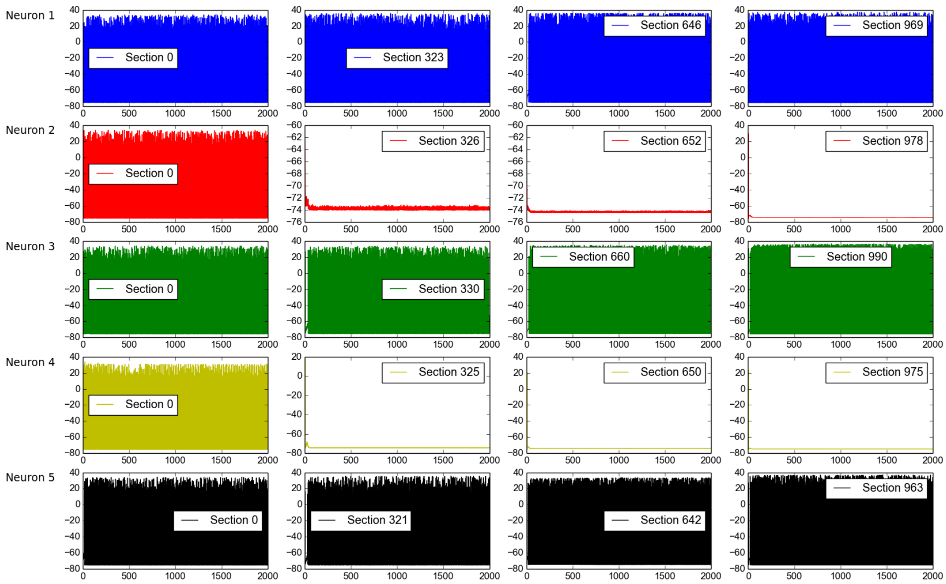

Figure 5: Compartmental activity in a CSTMD1 neuron simulation showing inhibitory effects

We also extended the module to be able to run several simulations while given different inputs from the ESTMD module so as to observe if there is any evidence that our CSTMD1 module, when presented with two targets in the visual receptive field, would select one of them as we expected. However, after modifying all the possible parameters and trying different inputs, we observed that the CSTMD1 model is not displaying this selectivity. *TALK ABOUT GRAPH FROM WIEDERMAN PAPER*

The web client *(MULTI SIMULATIONS?????????????)* that we introduced supports running simulations of the CSTMD1 module. We transferred all the functionality as described above and thus the user is able to change all the significant parameters of the model and observe the resulting CSTMD1 neurons' behaviour as well as the behaviour of KPIs in the model.

4.4 Pattern Recognition

The purpose of the pattern recognition module is to discern recurring spike patterns within the output from the CSTMD1 module, with each pattern recognition neuron becoming selective to one pattern. Those patterns encode information about the velocity and direction of the target observed in the visual field of the dragonfly.

In order to model these neurons, we initially replicated experiments conducted by Masquelier et al. [3] [4]. These experiments showed that spike Response Model (SRM) leaky integrate-and-fire neurons could successfully recognise input patterns based on sample input generated from a Poisson process. A single of these neurons is able to successfully recognise a recurring pattern within background noise and a network of them is able to do so for multiple patterns. This behaviour is achieved by modulating the weights of the pattern recognition neuron's synaptic connections to its afferents using spike timing dependent plasticity (STDP). STDP uses Long Term synaptic Potentiation (LTP) to reinforce connections with afferents that fired shortly before the postsynaptic neuron, and Long Term Depression (LTD) to weaken those with afferents that fired shortly after. Given that the input patterns occur within random noise, STDP will favour those afferents that participate in the pattern, as every time the pattern manifests itself, they will consistently fire in a given order. Within 15 seconds of simulation time, the pattern recognition neuron becomes selective to the pattern, and continuously reinforces the connections of the afferents that fired slightly before it discharged. Hence with every manifestation of the pattern, the neuron is more likely to fire earlier within it, effectively signalling its beginning.

In order to allow for different pattern recognition neurons to become selective to different patterns, we followed Masquelier et al. (2009), connecting a network of pattern recognition neurons to the sample inputs, introducing inhibitory connections amongst the post-synaptic neurons. This allows for a single postsynaptic neuron to become selective to one pattern and to inhibit other postsynaptic neurons from becoming selective to that pattern, thus allowing them to bind to other patterns in the input.

In a more extended model, post-synaptic neurons can become selective to part of a specific pattern. Masquelier et al. (2009) showed that given a 50ms spike pattern, up to 3 different neurons can fire within a single pattern thus identifying the beginning, middle and end of it. By adjusting the level of the mutual inhibition the number of the firing post-synaptic neurons can be increased or reduced depending the goal of the simulation. This extended model, although not used in our end product, it initiated long discussions about its potential in this project. During the time that the action selection module was not progressing as expected, we considered the alternative of using the aforementioned model to create a multiple level pattern recognition mechanism. The goal would be to investigate the extent of information regarding the velocity of the target (encoded in the visual input) that could be properly decoded by this multilevel pattern recognition mechanism. Despite the fact that we could have obtained results valuable to the research community, we were well aware that such a decision would result in a significant deviation from our original goals. Therefore, we kept our focus on the action selection module instead.

Once the pattern recognition module was built, we extended it so that the neurons can be easily adapted to recognise input with varying properties such as average firing rate, number of afferents, frequency of pattern appearance, amongst others. This implementation is able to recognise patterns output from our CSTMD1 neurons and measures the effectiveness of the pattern recognition neurons by tracking key information such as true-positive, false-positive and true-negative spike incidences.

4.5 Action Selection

The function of this module is to convert the output of the pattern recognition neurons into an action for the dragonfly to take in order for it to pursue insects in its visual field. The main aspects we needed to consider were:

1. The simulation environment of the dragonfly
2. The embodiment of the dragonfly
3. The actions available to the dragonfly within its environment
4. How to convert the output of the pattern recognition neurons into actions

Since the purpose of this project is to provide a tool for studying dragonfly vision rather than prey capture, we decided that having a complicated 3D simulated environment for the dragonfly was unnecessary. Instead, we decided to use the original retina input as a basis for the 'environment' for the dragonfly, and simply add a red circle to the animation input that corresponds to the dragonfly's focal point. The focal point would be moved around within the 2D plane of the visual input by the action selection mechanism and the focal point being overlapping with a target in the visual field would represent the dragonfly having caught the target. The actions available to the dragonfly's focal point would be up, down, left and right within the visual field. Therefore, the output of the action selection module would be a video of the original animation with the position of the dragonfly's focal point superimposed to each frame. An example is shown in Figure 1.

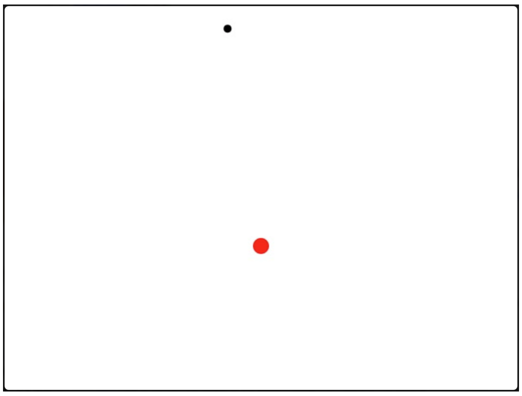

Figure 6: Simple example of action selection output. The red dot represents the dragonfly focal point and the black dot represents the target.

In the original proposal for the project, it was suggested that we use reward-modulated reinforcement learning to train the action selection mechanism. The idea of this is to have a mechanism that can be used to train the dragonfly to choose the actions that maximize its reward, which is administered when it catches a target.

We decided to implement a network of four action selection neurons, each representing one direction of movement (up, down, left and right). These neurons would take as input the spike train output of the pattern recognition neurons, via synapses with all-to-all connections between the two groups of neurons. The weights of the synapses would be initially randomised and the changes in weights would be dynamically governed by spike-timing-dependent plasticity (STDP, the same mechanism as the pattern recognition neurons) but modulated by a reward-modulator, which we label as dopamine but in reality could be another chemical. The way this works is that the change in weights determined by STDP is replaced with an eligibility trace that decays over time and the weights only change when dopamine is present in the system, by multiplying the eligibility trace of each synapse with the level of dopamine present. The dopamine is given a boost whenever the dragonfly focal point is within a certain distance of a target. The conceptual idea is that the synapses that contribute to catching the target are strengthened. The actual model used for the neurons is integrate-and-fire, which is a relatively simplistic but efficient model. The equations that govern the behaviour of the neurons and synapses are shown below:

Integrate-and-fire neurons:

$$\frac{dv}{dt} = \frac{g_e(E_e - v_r) + E_l - v}{\tau_m}$$

$$\frac{dg_e}{dt} = -\frac{g_e}{\tau_e}$$

where v is the membrane potential, g_e is the synaptic conductance, $E|e$ and E_l are reverse potentials, v_r is the resting potential and τ_m is a time constant.
Reward-modulated Synapses with STDP:

$$\frac{dw}{dt} = c \cdot Dop$$

$$\frac{dDop}{dt} = -\frac{Dop}{\tau_{Dop}}$$

$$\frac{dc}{dt} = -\frac{c}{\tau_c}$$

$$\frac{dApre}{dt} = -\frac{Apre}{\tau_{pre}}$$

$$\frac{dApost}{dt} = -\frac{Apost}{\tau_{post}}$$

where w is the weight of the synapse, c is the eligibility trace, Dop is the level of dopamine, $dApre$ and $dApost$ are variables that govern the STDP and the τs are time constants. When the pre-synaptic neuron fires, the following updates are executed:

$$ge = ge + w$$

$$Apre = Apre + Apre_{step}$$

$$c = c + Apost$$

When the post-synaptic neuron fires, the following updates are executed:

$$Apost = Apost + Apost_{step}$$

$$c = c + Apre$$

where $Apre_{step}$ and $Apost_{step}$ are constants.

Finally, at each time step, the distance between dragonfly's focal point (which is initialised to some position at the start) and the nearest target is calculated. If the distance is within a preset distance, a dopamine boost is administered to the system. The movement of the dragonfly is calculated by translating the firing rate of each of the four neurons into the velocity of the focal point in the corresponding direction. The overall velocity and hence distance travelled for the time step was averaged between the four and the dragonfly position updated.

It is possible to run the action selection part in training mode, where the weights can change, or in fixed mode where you give it trained, fixed weights to simply run a simulation.

The model was implemented in Python using a neuron simulator package called Brian 2 (CITATION). We used this as it is a flexible simulator and easier to use than NEURON (which was used for the CSTMD), since we are only modelling point neurons in this module, not a multi-compartmental model.

The output of the action-selection module is twofold:

1. A video of the original animation with the dragonfly focal point position superimposed to each frame. This way you can see if the dragonfly is 'chasing' the targets or not.
2. A graphical display of the behaviour of variables of the model during the simulation, including weights, firing rates, the dopamine level and a raster plot.

In the web client, the user is able to change the parameters of the model and observe how the resulting dragonfly's behaviour, as well as the behaviour of variables in the model.

4.6 Web Client

The web client is designed to be a simple interface through which simulations for each of the modules can be run and automated, both separately and jointly. This graphical user interface interacts with each module's API and provides the functionality needed to save and access the results of every experiment.

In order to maintain consistency with the rest of the project, we decided to use a web framework for Python that could easily connect with each of our modules. The basic requirements were that the client could be easily deployed on a local environment for testing purposes, but also able to serve multiple users concurrently when deployed in production. We chose Bottle (http://bottlepy.org) as it is a lightweight web framework with a built-in HTTP development server that would address the first requirement out-of-the-box. It can also be paired with Nginx (http://wiki.nginx.org/) through uWSGI (https://uwsgi-docs.readthedocs.org/en/latest/). By using Nginx, a high-performance HTTP server on top of uWSGI, a full stack interface between web frameworks and web servers, our web client can be deployed in production, providing load-balanced high-availability.

Given that running our modules' simulations is computationally expensive, it was imperative that the output and results of each run be saved in a persistent store. As shown in figure X, our project consists of five modules connected sequentially, with the output of the each module in the sequence serving as input to the next module. By saving the output of each module during a simulation, we would be able to perform multiple runs of the next module in the sequence without rerunning the previous modules. Additionally, by saving the data generated by each run, the web client can provide a view of the results for analysis by simply querying the database instead of generating them from scratch.

We chose MongoDB, a documented-oriented database, as a data store as it provides a fast, scalable solution that does not require strict design decisions in advance. As we developed our product, MongoDB's dynamic schemas allowed us to modify our objects without having to spend considerable time fixing compatibility issues. Additionally, MongoDB documents follow a JavaScript Object Notation (JSON)-like structure, which mirrors the structure of our Python objects, making it straight forward to understand what fields in our stored documents correspond to what properties of our Python classes. [DOCUMENT] [OBJECT]

Animation Object as stored in MongoDB

```
{

  "width": 640,
  "height": 480,
  "num_frames": 50,
  "background_id": 552d618629750413075fde0d,
  "description": "Sample animation.",
  "frames_per_second": 5,
  "targets": [
             { 'color': 'rgb(20,97,107)',
               'velocity': '5',
               'velocity_vector': ['1', '2'],
               'type': '1',
               'start_pos': ['1', '2'],
               'frames': '50',
               'size': '1' },

             { 'color': 'rgb(45,86,55)',
               'velocity': '2',
               'velocity_vector': ['2', '5'],
               'type': '2',
               'start_pos': ['10', '3'],
               'frames': '40',
               'size': '3' }
    ]
}
```

Animation Object constructor in Python

```python
def __init__(self,

width= 640,
height= 480,
num_frames= 50,
background_id= "552d618629750413075fde0d",
description= "Sample animation.",
frames_per_second= 5,
targets=[
         a.Target(color=rgb(20,97,107),
                 velocity=5,
                 velocity_vector=[1,2],
                 type=1,
                 start_pos=[1,2],
                 frames=50,
                 size=1),

         b.Target(color=rgb(45,86,55),
                 velocity=2,
                 velocity_vector=[2,5],
                 type=2,
                 start_pos=[10,3],
                 frames=40,
                 size=3)]):

self.targets = targets
self.width = width
self.height = height
self.background_id = background_id
self.num_frames = num_frames
self.frames_per_second = frames_per_second
self.description = description
```

Bottle has a built-in templating engine that enhances HTML with a thin layer of Python that can be inserted as both as inline and embedded snippets. These templates also allow the server to pass an object to the view, which can then be accessed using straightforward Python dictionary syntax.

```html
<!DOCTYPE html>
<html>
% include('head.tpl', title="Pattern Recognition Simulation")
<body>
% include('header.tpl')
<div class="container">
```

Separate the templates into reusable sections/blocks of HTML code that can be included dynamically in other templates.

```
<!-- Tab Panes -->
<div class="tab-content">
    % for i in range(len(simulation['potential_plots'])):
    % p_plot = simulation['potential_plots'][i]
    <div role="tabpanel" class="tab-pane" id="p{{i + 1}}">
        {{!p_plot}}
    </div>
    %end
</div>
```

Iterate through an array passed to the view and access the relevant fields in each iteration.

```
<tr>
    <td>True Positives</td>
    <td>{{'%.2f' % neuron['spike_info'][0][0]}}%</td>
    <td>{{'%.2f' % neuron['spike_info'][1][0]}}%</td>
    <td>{{'%.2f' % neuron['spike_info'][2][0]}}%</td>
    <td>{{'%.2f' % neuron['spike_info'][3][0]}}%</td>
</tr>
```

Format and access the field of an object passed to the view.

4.7 Problems To Bo Solved

4.8 Software Engineering Techniques

4.9 Division of Tasks

5 Group Work

This section describes the division of the total workload among the group members. Flexibility and adaptation were key in the successful division of the tasks. Juan Carlos Farah assumed the position of the team leader due to his previous working experience in technology related projects.

5.1 Initial Division

Initially the team was divided into two subgroups, the action selection and the pattern recognition group. The original estimation was that the action selection task would be much easier than the pattern recognition as for the former, relevant third party code was supplied. Therefore Christos Kaplanis along with Ioannis Kasidakis were assigned to the action selection group and Juan Carlos Farah, Erik Grabljevec and Panagiotis Almpouras were assigned to the pattern recognition group.

5.2 Reorganising to meet targets

As mentioned earlier in the report, the CSTMD neuron of the action selection part did not behave as expected. With the addition of the ESTMD neuron and the web client a reform was required to ensure that the project would progress as smoothly as possible. Erik Grablevec was tranferred to the action selection group and along with Christos Kaplanis started working on the ESTMD neuron. Ioannis Kasidakis kept working on the CSTMD neuron trying to solve the issues that had arisen. Juan Carlos Farah started working on the web client. Initially he created the general structure and later in cooperation with each member he connected the individual modules to the web client and with each other. Panagiotis almpouras continued working on the pattern recognition module finalising its dessign and development.

Later on, a new reorganisation was required. Christos Kaplanis and Erik Grablevec after having completed the ESTMD neuron, they were respectively assigned to the re-enforcement learning and the target animation.

5.3 Reports and Testing

With the reports and testing a general strategy was decided early on. Each member would be responsible for providing unit tests for the components he worked on. Same for the report each member should write the part for the report which was most relevant to the part he worked on. The more generic parts of each report were assigned the member(s) that were least busy at the time before the submission.

6 Final Product

From beginning till completion of this project a lot of hard work was put on it. The complex nature of the field of neuroscience as a whole significantly increased the overall level of difficulty of the project. Our goal was not just to replicate the neural processes occurring when the dragonfly preys but to create a tool that models them in the most realistic way possible and that provides helpful metrics for the analysis of those processes.

For most parts we accurately estimated the level of difficulty so that we can properly allocate resources on. Some however proved more challenging than we originally expected. This section summarises the goals that we met, the targets that we could only partially complete, the reasons why we could not fully address some of the issues that arose during the project as well as further development that could prove fruitful in the future.

6.1 Goals Met

As can be seen in the Specification section of this report we managed to fully complete most of the targets that we set. In fact, we managed to complete all the targets set not only as minimum requirements (Stage 1) but also for the expected implementation (Stage 2). More specifically, we managed to:

1. Create an animation tool that allows the generation of input for visual processing. (Minimum Requirement)
2. Build a model for the ESTMD neuron present between the retina and the actual CSTMD neurons of a real dragonfly. The function of this neuron is, given visual input, to isolate small targets from a potentially noisy background. (Minimum Requirement)
3. Build a layer of pattern recognition neurons that can be trained in an unsupervised manner to identify spike patterns within a noisy background. (Minimum Requirement)
4. Integrate the visual processing and pattern recognition system to detect patterns within the CSTMD output and add a simple action selection mechanism. (Minimum Requirement)
5. Develop a web client to provide an interface and key metrics for each module and for the system as a whole. (Expected Implementation)
6. Create an animation for the dragonfly agent to visualise the results of a simulation. (Expected Implementation)
7. Enhance the action selection mechanism to control the agent within the environment (Expected Implementation)
8. Improve the usability and features of the web client (Possible Exentions)

6.2 Partially completed tasks

The task that proved to be the most challenging aspect of the project was the development of the CSTMD neuron. It is observed that the CSTMD neuron when provided with multiple stimuli, it spikes depending only on one of them thus providing a selection mechanism [?]. In the beginning of the project we were provided with third party code that replicated that process but was tested only for a specific type of input. With the visual input that was key to our project however the CSTMD neuron did not show the expected behaviour and it failed to provide a robust selection mechanism. Despite our constant efforts to make it work, the time constraints of this project along with the high complexity of the other modules made it infeasible to fully meet this target. Therefore the target:

Achieve CSTMD target selection through experimentation with various parameters and connections with the ESTMD neuron (Possible Extensions) was only partially completed.

6.3 Non-implemented tasks

Of all the goals that were set there was only one we did not get the chance to work on:
Implement the agent in a quadcopter drone (Possible Extensions). The idea behind this goal was to integrate the completed agent to an actual quadcopter with embedded cameras that would allow it to react in real time to stimuli. That however would require very fast completion of the whole system or extended period of time to work on this project. What is more, the required processing of the visual input within each simulation is so computationally expensive that made it impossible to implement a real time system with the existing hardware. Therefore this goal could not be met.

6.4 Future Development

This project can definitely motivate future development. Even as an individual module, the CSTMD neuron could prove very useful to the science community. We may have not been able to model its behaviour properly, we did however observe that the existing tools regarding this module are insufficient and allow for plenty of space for improvement. What is more, if the system gets fully implemented, integrating it to a quadcopter with powerful processors could lead to numerous interesting applications.

6.5 Project Evaluation

All in all, we feel that we can consider this project a successful one. That is not only because we manage to meet most of the goals that we set despite their high level of difficulty but also because it was a great experience that every member of this team regards as valuable. There was a steep learning curve throughout the project and every member got the chance to familiarise themselves with several tools that have both generic and specific(???) applications. Team work was key to the success of this project and we are very satisfied that we managed to properly cooperate in a flexible and efficient way.

6.6 Testing & Results

References

[1] Python Software Foundation. *Python Language*.

[2] Zafeirios Fountas and Klaus Stiefel. Neuromorphic Engineering Workshop, 2013.

[3] T. Masquelier, R. Guyonneau, and S.J. Thorpe. Spike timing dependent plasticity finds the start of repeating patterns in continuous spike trains. *PLoS One*, 2008.

[4] T. Masquelier, R. Guyonneau, and S.J. Thorpe. Competitive STDP-based spike pattern learning. *Neural Comput.*, 2009.

[5] Mathworks. *Matlab & Simulink*.

[6] Misc. Dragonfly. `http://en.wikipedia.org/wiki/Dragonfly`.

[7] Robert M Olberg. Visual control of prey-capture flight in dragonflies. *Current Opinion in Neurobiology*, pages 1–5, 2011.

[8] Steven Wiederman. A Neurobiological and Computational Analysis of Target Discrimination in Visual Clutter by the Insect Visual System. (September), 2008.

[9] Steven Wiederman and David O'Carroll. Selective Attention in an Insect Visual Neuron. *Current Biology*, 2013.

A Logbook

B Minutes of Group Meetings

Logged by: Panagiotis Almpouras

This is a summary of the meeting minutes recorded throughout the project.

16 January 2015

Attendance

1. Juan Carlos Farah (JCF)
2. Christos Kaplanis (CK)
3. Erik Grabljevec (EG)
4. Panagiotis Almpouras (PA)
5. Ioannis Kasidakis (IK)
6. Zafeirios Fountas (ZF)
7. Pedro Martnez Mediano(PMM)

Summary

This meeting was mainly focused on dscussing the way the group would tackle administrative tasks. The decision about the split of the groups was finalised and initial tasks (mainly background reading) was assigned to the two groups. The members were split in a group of three (JCF, EG, PA) that would focus on the pattern recognition and a group of two (CK, IK) that would focus on the action selection.

21 January 2015

Attendance

1. Juan Carlos Farah (JCF)
2. Christos Kaplanis (CK)
3. Erik Grabljevec (EG)
4. Panagiotis Almpouras (PA)
5. Ioannis Kasidakis (IK)
6. Zafeirios Fountas (ZF)
7. Pedro Martnez Mediano(PMM)

Summary

This meeting was mainly focused on discussing questions we had after reading all the relevant papers as well as what should be the next steps. A joint meeting took place after which, separate group meetings were conducted with each relevant supervisor to discuss in more detail about the specifics of the upcoming tasks.

28 January 2015

Attendance

1. Juan Carlos Farah (JCF)
2. Christos Kaplanis (CK)
3. Erik Grabljevec (EG)
4. Panagiotis Almpouras (PA)
5. Ioannis Kasidakis (IK)
6. Zafeirios Fountas (ZF)

7. Pedro Martnez Mediano(PMM)

Summary

This meeting was mainly focused on the progress of the two groups so far. Most tasks (regarding replicating work that has been done by researchers so that members get comfortable with the concepts and methods) were on track and progressing as expected. A decision was made on the general structure of the first report (due on 06/02/2015) and each member was assigned a part of the report that they would have to prepare before the next meeting.

04 February 2015

Attendance

1. Juan Carlos Farah (JCF)
2. Christos Kaplanis (CK)
3. Erik Grabljevec (EG)
4. Panagiotis Almpouras (PA)
5. Ioannis Kasidakis (IK)
6. Zafeirios Fountas (ZF)
7. Pedro Martnez Mediano(PMM)

Summary

This meeting was mainly focused on the first report. With the input of the supervisors and the slides provided by the course instructor Dr. Fidelis Perkonigg the final structure of the report was decided and a soft deadline was set for completion, to allow for a few last amendments before the actual submission on the 06/02/2015.

11 February 2015

Attendance

1. Juan Carlos Farah (JCF)
2. Christos Kaplanis (CK)
3. Erik Grabljevec (EG)
4. Panagiotis Almpouras (PA)
5. Ioannis Kasidakis (IK)
6. Zafeirios Fountas (ZF)
7. Pedro Martnez Mediano(PMM)

Summary

This meeting was mainly focused on the progress and issues of each group. The pattern recognition group (JC, EG, PA) had no issues to report and made progress according to plan. The action selection group (CK, IK) upon completion of the initial tasks realised they need an extra component to act as an intermediate between the stimuli input and the action selection neurons. The main issue was the isolation of the small moving targets from much larger objects that may be moving in the background. The action selection group set a goal for next week to identify the most feasible solution by conducting deep background research on the topic.

18 February 2015

Attendance

1. Juan Carlos Farah (JCF)
2. Christos Kaplanis (CK)
3. Erik Grabljevec (EG)
4. Panagiotis Almpouras (PA)
5. Ioannis Kasidakis (IK)
6. Zafeirios Fountas (ZF)
7. Pedro Martnez Mediano(PMM)

Summary

This meeting was mainly focused on the progress and issues of each group. The pattern recognition group (JC, EG, PA) had no issues to report and made progress according to plan. The pattern recognition was providing accurate results and was modularised so that it could run independently given a proper input.
The action selection group (CK, IK) upon completion of the initial tasks realised they need an extra component to act as an intermediate between the stimuli input and the action selection neurons. The main issue was the isolation of the small moving targets from much larger objects that may be moving in the background. The action selection group set a goal for next week to identify the most feasible solution by conducting deep background research on the topic.

25 February 2015

Attendance

1. Juan Carlos Farah (JCF)
2. Christos Kaplanis (CK)
3. Erik Grabljevec (EG)
4. Panagiotis Almpouras (PA)
5. Ioannis Kasidakis (IK)
6. Zafeirios Fountas (ZF)
7. Pedro Martnez Mediano(PMM)

Summary

This meeting was mainly focused on the action selection mechanism. The action selection group managed to identify a viable solution to the issues mentioned in last week's meeting. An intermediate mechanism should be constructed to approximate the function of the ESTMD neuron. The function of the mechanism would be to pre-process the visual input to isolate small targets. Since the pattern recognition group had almost completed its task one member (EG) was transferred to the action selection group to assist with the development of ESTMD . A web client was also deemed as a good addition to the project to provide a user friendly interface that connects all the modules together but also allows for independent use of each. The pattern recognition group was further split. One member would focus on the web client (JCF) and the other (PA) would continue working on the pattern recognition mainly creating unit tests to ensure that future changes would not create unspotted issues that affect the robustness and accuracy of the module.

4 March 2015

Attendance

1. Juan Carlos Farah (JCF)

2. Christos Kaplanis (CK)
3. Erik Grabljevec (EG)
4. Panagiotis Almpouras (PA)
5. Ioannis Kasidakis (IK)
6. Zafeirios Fountas (ZF)
7. Pedro Martnez Mediano(PMM)

Summary

This meeting was mainly focused on the web client and the ESTMD neuron. A skeleton version of the web client was created by (JCF) and the next step would be to connect the completed pattern recognition module to the web client. Progress was reported on the ESTMD, however the intermediate mechanism revealed some unspotted issues with the CSTMD neuron that is responsible for selecting a single target if multiple stimuli are provided. The action selection group was divided into two subgroups. Two members (CK, EG) would continue working on the ESTMD and the third member (IK) would focus on the CSTMD to analyse the erroneous behaviour and identify potential solutions. The structure of the second report (due on 13/03/2015) was also discussed.

11 March 2015

Attendance

1. Juan Carlos Farah (JCF)
2. Christos Kaplanis (CK)
3. Erik Grabljevec (EG)
4. Panagiotis Almpouras (PA)
5. Ioannis Kasidakis (IK)
6. Zafeirios Fountas (ZF)
7. Pedro Martnez Mediano(PMM)

Summary

This meeting was mainly focused on the second report. With the input of the supervisors and the slides provided by the course instructor Dr. Fidelis Perkonigg the final structure of the report was decided and a soft deadline was set for completion, to allow for a few last amendments before the actual submission on the 13/03/2015.

18 March 2015

Attendance

1. Juan Carlos Farah (JCF)
2. Christos Kaplanis (CK)
3. Erik Grabljevec (EG)
4. Panagiotis Almpouras (PA)
5. Ioannis Kasidakis (IK)
6. Zafeirios Fountas (ZF)
7. Pedro Martnez Mediano(PMM)

Summary

This meeting was mainly focused on the web client and the ESTMD neuron. The action selection module was successfully connected to the web client and some prefixed simulations could be run online. Next step would be to add as many functionalities as possible to the web client before the completion and connection of the other modules. Progress was reported on the ESTMD development (CK,EG). No progress was reported on the CSTMD neuron (IK) but a lot of possible ways to address the issue were discussed and would be looked into before the next meeting.

25 March 2015

No meeting was held this week due to examinations.

1 April 2015

Attendance

1. Juan Carlos Farah (JCF)
2. Christos Kaplanis (CK)
3. Erik Grabljevec (EG)
4. Panagiotis Almpouras (PA)
5. Ioannis Kasidakis (IK)
6. Zafeirios Fountas (ZF)
7. Pedro Martnez Mediano(PMM)

Summary

This meeting was mainly focused on planning the group project activities to take place during the exam preparation period. Members mutually agreed to focus on exams and work on the group project one day a week for the following weeks. Slower but steady progress was expected. The CSTMD neuron proving much more challenging than originally expected was the only section for which an accurate estimate of the completion date could not be made. For that reason the simpler but similar mechanism of the point neurons was decided to be the last resource in case the CSTMD could not function properly.

8 April 2015

No meeting was held this week due to Easter holiday.

15 April 2015

Attendance

1. Juan Carlos Farah (JCF)
2. Christos Kaplanis (CK)
3. Erik Grabljevec (EG)
4. Panagiotis Almpouras (PA)
5. Ioannis Kasidakis (IK)
6. Zafeirios Fountas (ZF)
7. Pedro Martnez Mediano(PMM)

Summary

This meeting was mainly focused on discussing the progress of the development of the several modules. The web client was progressing as expected and as soon as each module was completed it would get connected to the web client. The ESTMD was progressing as expected and completed. The CSTMD still could not progress. The target animation along with the re-enforcement learning mechanism were the next things to be tackled. The target animation was assigned to (PA) and (EG), the re-enforcement learning was assigned to (CK). (JCF) would continue working on the web client and (IK) on the CSTMD and the point neurons.

22 April 2015

Attendance

1. Juan Carlos Farah (JCF)
2. Christos Kaplanis (CK)
3. Erik Grabljevec (EG)
4. Panagiotis Almpouras (PA)
5. Ioannis Kasidakis (IK)
6. Zafeirios Fountas (ZF)
7. Pedro Martnez Mediano(PMM)

Summary

This meeting was mainly focused on discussing the progress of the development of the several modules. All modules apart from CSTMD were progressing as expected. The CSTMD showed some progress. All members mutually agreed not to work on the group project during the following two weeks as the final exams where taking place during that period.

29 April 2015

No meeting was held this week due to examinations.

6 May 2015

No meeting was held this week due to examinations.

8 March 2015

Attendance

1. Juan Carlos Farah (JCF)
2. Christos Kaplanis (CK)
3. Erik Grabljevec (EG)
4. Panagiotis Almpouras (PA)
5. Ioannis Kasidakis (IK)
6. Zafeirios Fountas (ZF)
7. Pedro Martnez Mediano(PMM)

Summary

This meeting was mainly focused on creating a schedule for the activities to take place during the last week before the submission. (JCF) would work with each member to make the final connection of all the modules to the web client. (IK) would finish the CSTMD neuron asap to test its behaviour. (PA) would start working on the report, creating the main structure and writing any part that can be written without the input of another member (mainly Introduction, Specification, Group Work, Appendix). A soft deadline was set for completion of the report on the 12/04/2015, to allow for a few last amendments before the actual submission on the 15/04/2015.

13 May 2015

Attendance

1. Juan Carlos Farah (JCF)
2. Christos Kaplanis (CK)

3. Erik Grabljevec (EG)
4. Panagiotis Almpouras (PA)
5. Ioannis Kasidakis (IK)
6. Zafeirios Fountas (ZF)
7. Pedro Martnez Mediano(PMM)

Summary

This meeting was mainly focused on the final report. With the input of the supervisors and the materials provided by the course instructor Dr. Fidelis Perkonigg the contents of the report were finalised. A brief discussion on the presentation took place. The presentation would be prepared after the submission of the final report on the 15/04/2015 as the date of the presentation was set to be the 19/05/2015.

C Detailed Work Breakdown

Member	Task	Time
Juan Carlos Farah	Pattern Recognition Module	6.5 weeks
	Web Client	4.5 weeks
	Testing	0.5 weeks
	First Report	0.5 week
	Second Report	0.5 week
	Final Report	1.0 week
Panagiotis Almpouras	Pattern Recognition Module	8.0 weeks
	Testing	2.0 weeks
	First Report	0.5 week
	Second Report	0.5 week
	Final Report	1.5 weeks
Christos Kaplanis	Action Selection Module	7.5 weeks
	Re-enforcement Learning	3.5 weeks
	Testing	0.5 weeks
	First Report	0.5 week
	Second Report	0.5 week
	Final Report	1.0 week
Erik Grabljevec	Pattern Recognition Module	3.0 weeks
	Action Selection Module	4.0 weeks
	Target Animation	3.5 weeks
	Testing	0.5 weeks
	First Report	0.5 week
	Second Report	0.5 week
	Final Report	1.0 week
Ioannis Kasidakis	Action Selection Module	7.5 weeks
	CSTMD module	3.5 weeks
	Testing	0.5 weeks
	First Report	0.5 week
	Second Report	0.5 week
	Final Report	1.0 week