AN IMPROVED PATH INTEGRATION MECHANISM USING NEURAL FIELDS WHICH IMPLEMENT A BIOLOGICALLY PLAUSIBLE ANALOGUE TO A KALMAN FILTER

by

Warren A. Connors

Submitted in partial fulfillment of the requirements
for the degree of Master of Computer Science

at

Dalhousie University
Halifax, Nova Scotia
February 2013

DALHOUSIE UNIVERSITY

FACULTY OF COMPUTER SCIENCE

The undersigned hereby certify that they have read and recommend to the Faculty of Graduate Studies for acceptance a thesis entitled "AN IMPROVED PATH INTEGRATION MECHANISM USING NEURAL FIELDS WHICH IMPLEMENT A BIOLOGICALLY PLAUSIBLE ANALOGUE TO A KALMAN FILTER" by Warren A. Connors in partial fulfillment of the requirements for the degree of Master of Computer Science.

Dated: February 22, 2013

Supervisor:

Dr. Thomas Trappenberg

Readers:

Dr. John Fawcett

Dr. Dirk Arnold

DALHOUSIE UNIVERSITY

DATE: February 22, 2013

AUTHOR: Warren A. Connors

TITLE: AN IMPROVED PATH INTEGRATION MECHANISM USING NEURAL FIELDS WHICH IMPLEMENT A BIOLOGICALLY PLAUSIBLE ANALOGUE TO A KALMAN FILTER

DEPARTMENT OR SCHOOL: Faculty of Computer Science

DEGREE: M.C.Sc. CONVOCATION: May YEAR: 2013

Signature of Author

Table of Contents

List of Tables

List of Figures

Abstract

Interaction with the world is necessary for both animals and robots to complete tasks. This interaction requires a sense of self, or the orientation of the robot or animal with respect to the world. Creating and maintaining this model is a task which is easily maintained by animals, however can be difficult for robots due to the uncertainties in the world, sensing, and movement of the robot. This estimation difficulty is increased in sensory deprived environments, where no external, inputs are available to correct the estimate. Therefore, self generated cues of movement are needed, such as vestibular input in an animal, or accelerometer input in a robot. In spite of the difficulties, animals can easily maintain this model. This leads to the question of whether we can learn from nature by examining the biological mechanisms for pose estimation in animals. Previous work has shown that neural fields coupled with a mechanism for updating the estimate can be used to maintain a pose estimate through a sustained area of activity called a packet. Analysis of this mechanism however has shown conditions where the field can provide unexpected results or break down due to high accelerations input into the field. This analysis illustrates the challenges of controlling the activity packet size under strong inputs, and a limited speed capability using the existing mechanism. As a result of this, a novel weight combination method is proposed to provide a higher speed and increased robustness. The results of this is an increase of over two times the existing speed capability, and a resistance of the field to break down under strong rotational inputs.

This updated neural field model provides a method for maintaining a stable pose estimate. To show this, a novel comparison between the proposed neural field model and the Kalman filter is considered, resulting in comparable performance in pose prediction. This work shows that an updated neural field model provides a biologically plausible pose prediction model using Bayesian inference, providing a biological analogue to a Kalman filter.

List of Abbreviations and Symbols Used

AUV - Autonomous Underwater Vehicle

CANN - Continuous Attractor Neural Network

DNF - Dynamic Neural Field

DRDC - Defence Research and Development Canada

DVL - Doppler Velocity Log

GPS - Global Positioning System

INS - Intertial Navigation System

NF - Neural Field

NMDA - N-methyl-D-aspartate

PI - Path Integration

Acknowledgements

I would like to thank my colleagues at DRDC, for their encouragement and support throughout this. Without those discussions and support, this would not have been possible. I would also like to thank Dr. Thomas Trappenberg, who inspired me to consider biological solutions for non-biological problems. His guidance and encouragement has changed both my approach but also my interest in the field of Neuroscience. Finally, none of this work would have been possible without the support and encouragement of my bride Amy, and my family who have always supported my interests and encouraged me to continue pursuing those interests. Last but certainly not least, my deepest gratitude goes to my father, Walter J. Connors. His encouragement, pride and support has had an incalculable impact on who I am, and what I desire to achieve. - Thank you.

Chapter 1

Introduction

In both robots and animals, some level of interaction with the world is critical to complete useful tasks, and even to survive. In order to interact with the world however, methods for sensing and modeling the environment, planning an action, and executing the actions based on the planning is required. A task which animals complete regularly, such as navigation, is a complex task in robotics involving the development and maintenance of an internal model of the robot's state in the world, and updating this representation using information acquired through sensing the environment. This sensed information is integrated to maintain a model of self commonly referred to as pose [31]. This pose model represents the orientation and/or location of a robot or animal. Depending on the robot and task, this could be a simple model maintaining orientation, or a complex model including attributes such as the configuration of manipulators, or the individual pose of sensors [31]. This model is critical to interacting with the world as navigation requires a representation of pose to determine the steps required to achieve a goal.

The pose model can be difficult to acquire and maintain, due to the uncertainties introduced by the sensing of and interacting with the environment. Sources of this uncertainty include noisy and incomplete sensing of the environment, uncertainty from the changing environment itself, and uncertain movement of the robot in the environment. For robotics, probabilistic models such as the Bayes filter have been developed which take this uncertainty into account and update the model of pose with the most likely pose estimate [31]. An additional challenge to this is the case where sensing of the external environment is impossible or incomplete due to unreliability of the sensors, and factors in the environment itself. This requires the pose estimate to be updated using self-generated cues, such as accelerations or self sensed velocities,

without corrections from sensed world state such as visual input.

An example of this drawn from underwater robotics is maintaining a pose estimate with an Autonomous Underwater Vehicle (AUV), such as an Iver2 (Fig. 1.1). When operating on the surface, this vehicle can maintain a pose model through the use of the Global Positioning System (GPS), a series of satellites in low earth orbit that use differential positioning to determine a location for a receiver. This input is combined with acoustic sensors such as a Doppler Velocity Log (DVL), which uses acoustics to provide a set of along track and across track velocities. These vehicles are designed to work in the subsurface environment however, where GPS is not available. Without a GPS input, the pose problem is considerably more difficult, as the system must consider using the uncertain velocities from the DVL, in combination with a compass to generate a pose model for the vehicle [14],[10],[9]. This model is degraded, as it will suffer from drift due to uncertainty in the accelerations, and inaccuracies in the compass. In specific cases, such as the Iver2 vehicle, periodic surfacing is used to update this model and correct for the drift. This method however, assumes that surfacing is possible and safe for the vehicle.

Figure 1.1: One of the DRDC Iver2 lightweight autonomous underwater vehicles. When submerged, the vehicle uses dead reckoning, incorporating DVL and compass input to maintain an estimate on current positioning between GPS fixes when surfaced.

In 2009 and 2012, Defence Research and Development Canada (DRDC) conducted scientific research in Canada's high arctic to examine the use of AUV's for arctic surveillance and data collection [7]. This research involved the use of the Iver2 vehicle (Fig. 1.1), and DRDC's Arctic Explorer (Fig. 1.2), a much larger and more complex vehicle designed for prolonged periods, up to 4 days, of underwater navigation without external positioning updates. The Explorer was designed for a complex navigation

mission, to enter the water through an ice hole, travel for multiple days under the ice, an then return to a pre-programmed area for recovery or recharging. As these missions were specifically designed to be run under the arctic ice, observation updates such as GPS fixes were not available, therefore the vehicle was required to navigate without external references [7],[17]. The environment in the high arctic further complicated the vehicle's estimates of pose, as the magnetic environment is variable, causing unstable measurements from the compass. These unstable measurements render the integration, or dead reckoning, method of pose estimation ineffective when using the compass. To correct this issue, complex devices called Inertial Navigation Systems (INS) and gyrocompasses were used to integrate angular velocities and accelerations with an existing model to update a pose estimate [7],[14],[10],[9].

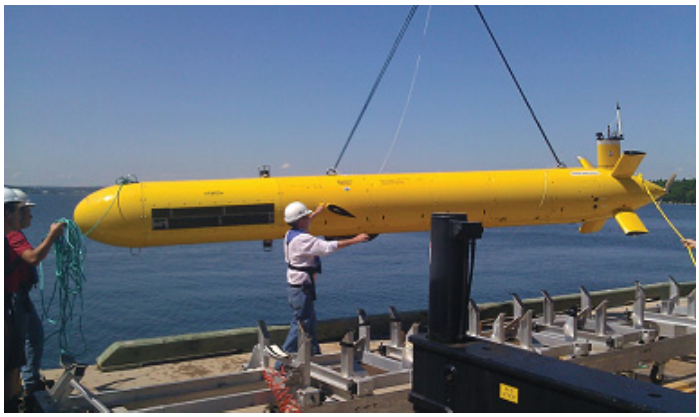


Figure 1.2: One of the DRDC Arctic Explorer autonomous underwater vehicles. When submerged, the vehicle uses an Inertial Navigation System to provide an estimate on current positioning between GPS fixes when surfaced.

The challenge of navigating in a sensory deprived environment is a situation faced by animals every day. Consider sensory deprived environments such as a dark room, or being a passenger in an automobile with closed eyes. Animals have the ability to maintain and update an estimate of pose without continual external input. This pose or direction is maintained through idiothetic or self-generated updates which are integrated to develop a new estimate of the current pose. These inputs, as in the robotics example above, are noisy and can provide conflicting evidence to a pose

model, however animals still maintain a reasonable estimate in sensory deprived environments. The ability for animals to successfully maintain this pose estimate is appealing for research as it not only provides more insight into biologically plausible methods to explain how animals integrate pose information, but also to find methods which can be applied to robotics, effectively learning robotic techniques from nature.

The ability for animals to maintain an estimate of pose suggests that a representation is maintained in the brain which is continually updated from visual or idiothetic queues. A continuous attractor neural network (CANN) can encode a continuous value such as a head direction, or in two dimensions, a position of the animal in a plane [11]. To provide such functionality however, the network must be updated both through visual input, and also through idiothetic queues such as angular velocities through the vestibular system [11]. A common model proposed for head direction [28],[11],[21] is through the use of a neural field, where each node in the continuum is tuned or sensitive to a specific direction. This tuning sensitivity, in concert with the dynamics of a CANN above, leads to the development of an area of localized activity, or an activity packet, that is located over a particular portion of the field which is sensitive to the current head location of the animal.

Neural fields using inhibition and excitation were originally introduced by Wilson and Cowan, as a mean field approximation of spiking neurons [35]. These fields, combined with basic anatomical accounts of the cortex result in a model with local excitation and long distance inhibition [36]. They argued that for complex cortical functions, for example sensor processing, information is represented in the cortex as large areas of activity, and the number of cells involved are too large to be tractable for individual analysis. Later work from Wilson and Cowan refined this model, considering cortical and thalamic tissue as a sheet containing a fully connected set of homogeneous, equally distributed inhibitory and excitatory neurons, with not only lateral, but recurrent connections [36]. This model supported the formation of localized activity caused by input stimulus, and modulated through recurrent excitation and inhibition. In this work, the dynamic activity of the field was considered, which

led to the introduction of the potential states of the field, showing transient activity and stable states. Through this work an inhomogeneous, stable, steady state was proposed as a possible mechanism for short term memory [36].

Amari later formalized the Wilson and Cowan model [1], and simplified it to a single layer model of a field, which contains both inhibitory and excitatory connections. His work further examined the five types of dynamics that arise from the field model. These five types of dynamics include (1) a monostable field where all activity dies out after removal of stimulus, (2) a monostable type field where the entire field becomes excited, (3) a bistable field where excitation of the nodes will spread without limit if the initial input stimulus is wide enough, (4) a bistable field were a localized packet of activity is formed or removed, where the area of activity moves to the area of maximum input stimulus, and (5) a spatially periodic excitation pattern. The first two cases were not considered in depth, as the goal was to consider the bistable cases, and the dynamics of the field that cause them. As shown by Amari [1], the level of inhibition in the system dictates the dynamics of the field. In cases of weak inhibition, the field is dominated by excitatory influences, and therefore becomes fully excited resulting in the activity packet growing to cover the field. In cases of high inhibition, the dynamics of the network is dominated by inhibition, therefore the activity of the field decreases and eventually dies out when external input is removed. The intermediate case is the one of interest for this analysis, where activity packet is maintained through recurrent excitation in the area of the field sensitive to the input after that input is removed. These areas are formally fixed point attractors, and allow encoding of a state or feature. Figure 1.3 illustrates the development and maintenance of an activity packet over time using a neural field. These characteristics provide a model for the proposed short term memory by Wilson and Cowan [35]. Taylor extended the work of Amari into two dimensions, formally showing the dynamics of the field and considering temporal aspects of the activity, with respect to the formation of activity packets, the characteristics of decay for the packets, and the temporal history of interaction between two bubbles [30].

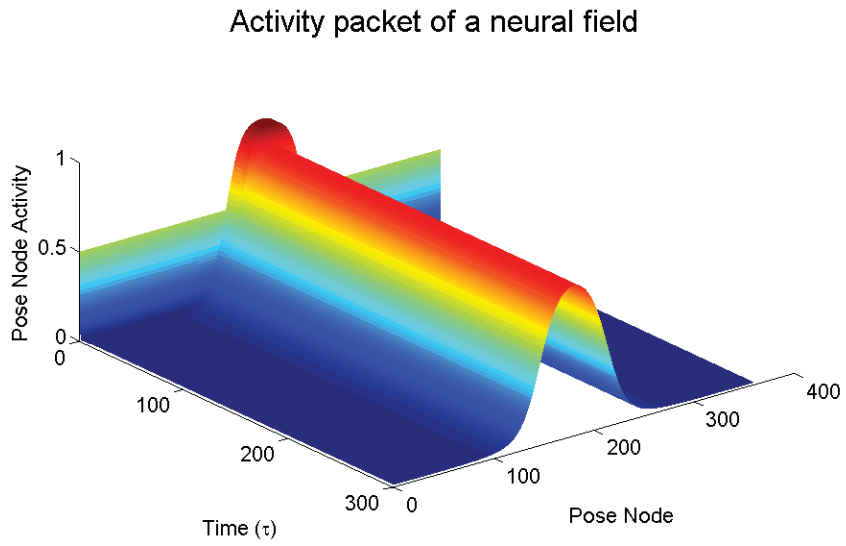


Figure 1.3: A stable localized area of activity in a neural field, called an activity packet. This activity is maintained through recurrent input in the excited nodes. The input was provided at t=1-10, and then removed, where the packet stabilized over the area of stimulus.

Neural fields are used extensively for modeling biological phenomena in areas including visual cortex [4], the superior colliculus in primates [33], spatial and episodic memory [19], cognition [22], visual working memory [5], and Hippocampus [39],[28],[27]. More recently, neural field models have been applied to non-biological systems in robotic sensing, motion, navigation [13], [40],[3] and cognitive robotics [40],[23].

Using the above model however, poses a challenge with respect to idiothetic input. Although visual input can be considered an external input to the field, self directed movement of the activity packet requires mechanisms which are not captured in the neural field model. To provide a stable solution in the neural field model, the activity packet must have symmetric excitatory input from neighbouring nodes [1], which allows stable formation of localized activity. To enable stable movement of the packet however, asymmetries in the activity must be introduced to provide a mechanism for movement.

Stringer et al. [28] have shown how asymmetric weights can be learned and how to apply these weights in combination with the neural fields to introduce a mechanism

for movement of the activity packet called path integration (PI). This was used to explain updating head direction and place representations from vestibular signals. This mechanism was also applied to motor control [29], however no experimental analysis of the behaviour with respect to the stability and robustness of a PI mechanism using neural fields was conducted.

An analysis of the PI mechanism proposed by Stringer et al. [28] was conducted as part of this work, specifically with respect to the movement speed of the activity packet under strong rotational inputs. This analysis illustrated specific limitations caused by the method of combining the field and PI weights. These limitations result in instability of the field under strong rotational input, and limited range of speed for the packet movement. As a result of these limitations, a novel method for combining symmetric and asymmetric weights is proposed to yield more robust PI over a larger range of rotational inputs, and a higher overall speed for the activity while maintaining the localized activity packet size. In addition, Stringer et al. [29] introduced a stabilization mechanism, referred to as NMDA-stabilization to prevent drifting of the activity packet under conditions of noisy or incomplete training of field weights. The effects of NMDA-stabilization on the PI are examined in more detail.

This updated PI model is compared to a Kalman filter, an optimal state estimation tool for linear systems [6],[24], to show the plausibility of neural fields as a biological implementation of a pose prediction model. As this method is being applied to implement a biologically plausible method analogous to a Kalman filter, the relationship between a Kalman filter using angular velocities from a gyro as control inputs will be examined. To illustrate this relationship, an experiment using a Kalman filter and the updated neural field model using PI with a correction step will be discussed. Here I argue that this mechanism provides, in essence, a biological Kalman filter.

The primary contributions of this work are the comparison of the neural field model to a Kalman filter, and the development of a new weight combination method for PI based on a robustness analysis. A detailed comparison between a Kalman filter and a neural field showing the similarities and differences between the two models

has not been considered previously. For PI, the proposed weight combination method is a modification of the existing weight combination methods which results in more stable control of the activity packet under a wide range of rotation inputs.

The remainder of this thesis is structured as follows. Chapter 2 will examine the structure and dynamics of a Kalman filter as well as the neural field with PI. These models will be compared to show analogous functions and concepts. Chapter 3 will illustrate the common goals and concepts of a neural field model using PI with a Kalman filter, through an experimental implementation for pose estimation. This experiment will compare the two models to integrate a series of compass headings with gyro input, which can be considered a non-biological analogue of integrating vestibular inputs using a neural field for head direction tracking in animals. Chapter 4 will outline the results of a robustness analysis on the PI mechanism proposed by Stringer et al. [28]. This chapter will propose modifications to the PI mechanism to allow for a higher level of stability at strong inputs, robust control of the activity packet, and address an inconsistency with the learning method. These modifications provide a more robust PI mechanism for stable movement and control of the activity packet over a wider range of rotational input. Chapter 5 provides conclusions on the updated neural field model, and the usefulness of the model in integrating both external and idiothetic input to provide a biologically plausible pose estimation.

Chapter 2

Neural Fields with Path Integration - Nature's Kalman Filter?

The problem of pose estimation through the integration of uncertain, or in some cases absent external inputs has been considered for both biological and non-biological systems as noted previously. Using the example illustrated in the introduction of this thesis, underwater navigation using dead reckoning or integration of noisy inputs from sensors such as accelerometers and Doppler velocity logs employs state estimation models such as Kalman filters to provide a reasonable estimate of pose, with consideration of the uncertainty that exists in the system. As animals and humans have a sense of direction in sensory deprived environments as well, this implies that a model of pose is maintained through idiothetic input which can be corrected when external sensory input is available. This chapter introduces the algorithm for the Kalman filter, examines the dynamic equations that define a neural field, builds on the neural field equations for PI functionality, and presents a comparison between the Kalman filter and neural fields using PI. This chapter further proposes a method where neural fields could be updated from external or non-idiothetic sensory inputs to provide a better estimate of pose, increasing the accuracy of the estimate through corrections, in the same manner as the Kalman filter.

2.1 Kalman Filters

A Kalman filter is a state estimation algorithm which uses a series of uncertain measurements, the uncertain results from motor commands, and the previous pose from time $t-1$ to estimate a new pose at time t. This filter was originally proposed by Rudolf Kalman in 1960 [6], and is commonly used in robotics [31],[8],[34], navigation

[18],[20], and signal processing [16],[31]. A Kalman filter is an implementation of a Bayes filter, requiring that the system observes the Markov property. This property states that the state at $t+1$ is only dependent on the state at t, and not any events preceding it. Furthermore, the Kalman filter places a linear assumption on the system such that the state transition probability must be linear in its arguments, meaning that the state transition must be a linear combination of the previous state with the results from motor commands and uncertainty modeled by added Gaussian noise. Furthermore the measurement probability must also be linear, again with uncertainly characterized by added Gaussian noise. When these constraints hold, Kalman filters are optimal filters for linear systems with Gaussian distributed noise [6],[31]. The Kalman filter has been extended for non-linear state estimation [25],[31],[8], and continuous time variants [2].

Kalman filters are characterized by a state estimate (μ_t) at time t, and and corresponding uncertainty characterized by a covariance (Σ_t). This filter uses a linear state transition, where the estimated state x_t is defined by,

$$\text{Motion model: } x_t = Ax_{t-1} + Bu_t + \xi \tag{2.1}$$

where A is the state transition constant, which relates the state at $t-1$ with the state at t, B is the control input constant, relating the control inputs u_t to the state, and ξ is the process uncertainty in the corresponding state transition. This process noise can be seen as modeling the uncertainty introduced into the system by the corresponding state transition. In practice, the constants A and B can be scalar constants or matrices, depending on the application. For example, in a one dimensional case these are constants, however in a two dimensional or higher case, these parameters are matrices representing the variance between the dimensions. Furthermore, observations are introduced into the system as a measurement which is also linearly combined with a uncertainty model. Measurements are defined by,

$$\text{Observation model: } z_t = C_t x_t + \delta \tag{2.2}$$

where C is the observation constant, which translates the current state to an expected observation, with uncertainty characterized by the additive noise specified by δ.

The Kalman filter, like the Bayes filter on which it is based, operates using a two step process, a prediction step and a correction step. These steps are outlined in the algorithm description in Algorithm 1. The prediction step computes the estimated belief of the current state based on the state at time $t-1$, and the motor commands that were used to translate the state, denoted by u_t. The predicted uncertainty is represented in the covariance estimate ($\bar{\Sigma}$), which is also updated during this stage based on the state transition constant (A) and uncertainty represented by the process noise, which is assumed to be Gaussian with a mean of 0 and covariance R. Following the prediction step is the correction step, where measurements are incorporated into the a priori predicted state to improve the pose prediction. In this step, the Kalman filter computes the Kalman gain, denoted by K_t, which determines the extent that the observation or measurement is taken into account when updating the state prediction. This gain is calculated by combining the estimated state covariance with the observation constant and additive Gaussian uncertainty on the measurement, with a mean of 0 and covariance Q. Once the Kalman gain has been computed, it is used to modify the state prediction by weighting the difference between the observation or measurement with the expected observation at time t. This difference, referred to as the innovation, is the effective correction used in the Kalman filter caused by the observation. The Kalman gain specifies the weighting that the innovation has, which can be thought of as the 'trust' that the filter places on either the apriori estimate, or the observed state. Once the state has been updated using the weighted innovation, the covariance on the final state estimate is computed and returned along with the corrected state estimate to be used in the next iteration of the Kalman filter.

The Kalman filter model has the effect of smoothing and integrating noisy sensor inputs and observations, to provide a more probable estimate of pose based on the integrated inputs. Figure 2.1 illustrates this effect, where noisy compass data was input into the Kalman filter, without accelerations input as motor command results. The smoothing effect of the Kalman filter is shown, where the estimated pose is less noisy. Kalman filters can be used for a simple one dimensional problem, such as the

Algorithm 1 The Kalman filter algorithm

KalmanFilter(μ_{t-1},Σ_{t-1},u_t,z_t)

/*Prediction Phase*/

/*create state estimate by combining the earlier state with motor commands*/

$\bar{\mu}_t \leftarrow A\mu_{t-1} + Bu_t$

/* update covariance on estimate by combining previous covariance with state trans and uncertainty */

$\bar{\Sigma}_t \leftarrow A\Sigma_{t-1}A^T + R$

/* Correction or update phase */

/* Compute the Kalman gain, by combining the predicted covariance with the observation model */

$K_t \leftarrow \bar{\Sigma}_t C^T (C\bar{\Sigma}_t C^T + Q)^{-1}$

/* Update the state through the innovation, modulated by the Kalman gain */

$\mu_t \leftarrow \bar{\mu}_t + K_t(z_t - C\bar{\mu}_t)$

/* Update the covariance using the Kalman gain */

$\Sigma_t \leftarrow (I - K_t C)\bar{\Sigma}_t$

return μ_t, Σ_t

pose estimate used in this thesis, or multi-dimensional applications, such as the INS mentioned in the Introduction. In a case such as an INS, the Kalman filter will use inputs from a gyro, accelerometers, DVL velocities, and GPS inputs where available. This data is all effectively smoothed and integrated to result in a multi-dimensional pose, including pitch, roll, heave, yaw and latitude/longitude position.

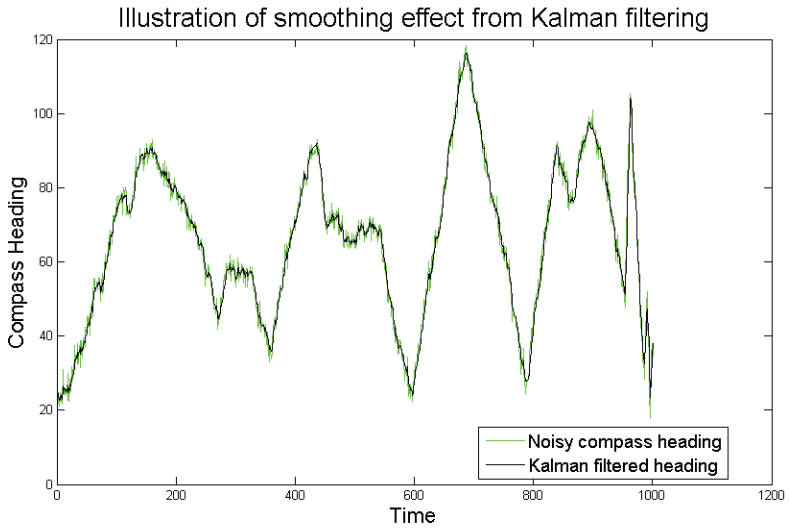


Figure 2.1: An illustration of Kalman filtering on a noisy compass heading. The resulting filtered headings are effectively smoothed versions of the noisy inputs.

2.2 Neural Field Dynamics

This section outlines the dynamics of the neural field model, examining the mechanisms for sustained activity in the field. This section will consider the structure of the field model, the dynamic equations which specify the model, and the training of kernel weights. Neural fields are described by the dynamic equation,

$$\tau \frac{\partial h(x,t)}{\partial t} = -h(x,t) + \int_y w(x,y) r(y,t) dy + I^{ext}(x,t), \tag{2.3}$$

where $h(x,t)$ is the activity of node x at time t, $w(x,y)$ is the synaptic weight between node x and node y, $r(y,t)$ is the firing rate of the field at location y, $I^{ext}(x,t)$ is the external input to the field at time t, and τ is the time constant of the system.

The result of this dynamic equation is a set of activities, which are then input into a sigmoid activation function,

$$r(x,t) = \frac{1}{1 + e^{\beta(h(x,t)-\alpha(x))}}, \tag{2.4}$$

where $r(x,t)$ is the firing rate of node x at time t, and β and α describe slope and the offset of how the field h is related to the firing rates r. Of specific interest are solutions of this system with stable localized activity packets that can encode and even memorize a pose of a system such as head directions or locations. These fields are often approximated with discrete pose cells that are labeled with index i, which in turn is related to the field location by

$$x = i\Delta x. \tag{2.5}$$

A one dimensional network of such pose cells is shown in Figure 2.3. The neural field model used in this work is organized as a ring, therefore periodic boundaries are defined at 0 and 2π. As noted in Chapter 1, the field depends on long distance inhibition and short distance excitation, therefore the dynamics of the field are dominated by the combination of the synaptic weights with the activity of the field. The short distance excitation serves to maintain the activity from decay, and the inhibition provides stability, preventing the packet from diffusing into the field [22]. These weights are specified by a distance dependent kernel function called the center surround kernel. This function can be modeler defined as long as the relationship between long distance inhibition and short distance excitation is maintained [32], however in practice this kernel function is typically described by a Gaussian radial basis function shifted by an inhibition constant, or a difference of Gaussians commonly known as a Mexican hat function. Figure 2.2 illustrates the difference in the weighting kernels. The difference in kernels provide different dynamics of the system. The Mexican hat kernel shown in Figure 2.2(b) will allow for the stable formation of multiple activity packets without competition between them, as long as the distance between is long enough to overcome the inhibitory influence and resting rate of the field. This can be seen by examining the kernel, where the inhibition of the field approaches zero,

therefore given a long enough distance, the amount of inhibition is very low or non-existent. The shifted Gaussian kernel as illustrated in Figure 2.2(a) however provides for averaging or competition between two pose hypotheses, depending on the distance separating the two bubbles. For short distances, the neural field may effectively merge the activation, resulting in a peak at an averaged location. For longer distances, the interaction of inhibition cause a competition, as the blanket inhibition provided to the field is consistent over all locations and strengthened by the rates or belief of the existing pose hypothesis [22]. For this thesis, the shifted Gaussian kernel is used, providing a single estimate of pose.

Of particular interest is the self-organization of the weights between these pose cells. These weights can be self organized through a learning technique called the Hebb rule or Hebbian learning. This rule associates the increase in synaptic weight or efficacy between two neurons with the consistent firing of pre-synaptic and post-synaptic nodes. Informally, this is known as the rule 'Neurons that fire together, wire together' [32]. This method provides a basic learning algorithm for neural networks, allowing for the adjustment of network weights according to activity.

As noted above, the choice of weight kernel used in this work is based on a shifted Gaussian radial basis function Fig. 2.2(a), shifted by a constant value to represent the inhibition in the system. This kernel is defined by,

$$w(x, y) = A_w(\tilde{w}(x, y) - C), \tag{2.6}$$

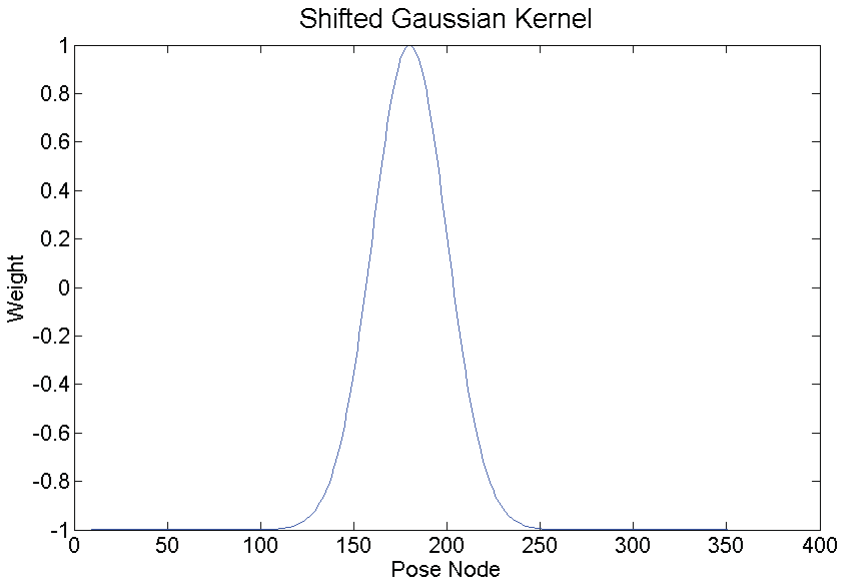
where C is a inhibition constant, A_w is a kernel scaling factor, The weights, $\tilde{w}(x, y)$, are learned through the Hebbian learning rule specified by,

$$\tilde{w}(x, y) = \int_0^{2\pi} r(x - x^{\mathrm{p}})r(y - x^{\mathrm{p}})\mathrm{d}x^{\mathrm{p}}, \tag{2.7}$$

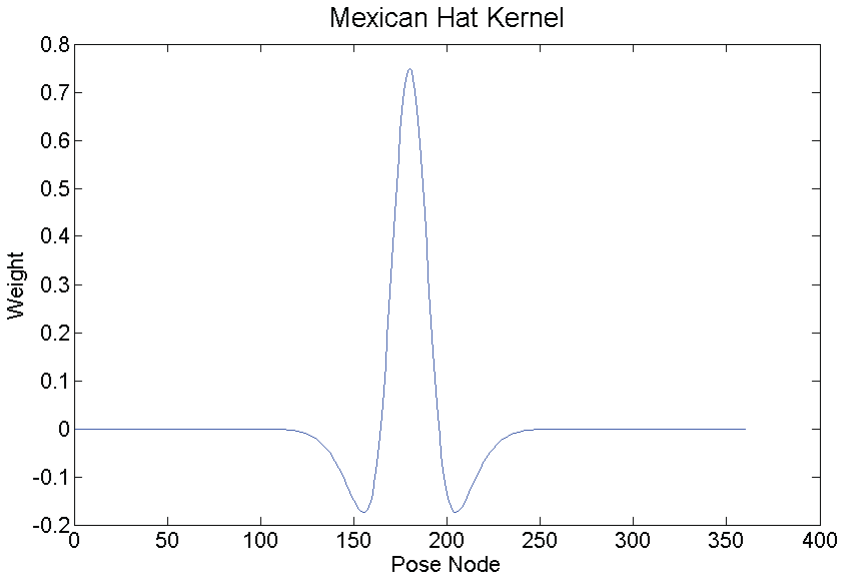
based on Gaussian radial basis function training patterns centered around the preferred direction, x^{p},

$$r(x - x^{\mathrm{p}}) = \mathrm{e}^{-(x-x^{\mathrm{p}})^2/2{\sigma_r}^2}. \tag{2.8}$$

where σ_r controls the width of the training patterns. These patterns allow the training of nodes in the field to encode a sensitivity to a set of features, for example direction



(a)



(b)

Figure 2.2: Comparison of the two commonly used kernel weight functions for neural fields, the shifted Gaussian (a), and the Mexican hat (b)

or pose, based on external stimulus during training. This preferred direction, x^{p} can be seen as the direction at which the node represents, or the direction 'tuning curve'. This sensitivity allows a node to be sensitive to a specific external input allowing the field to maintain a model of pose through recurrent input without continual external

stimulus.

The nodes and weights are continuous across the field, and can be formally considered in this work as a connected ring. To provide correct weight training in all directions with these conditions, periodic boundaries of the neural field are used by replacing the distance between x and x^{p} with,

$$|x - x^{\mathrm{p}}| \to \min(|x - x^{\mathrm{p}}|, 2\pi - |x - x^{\mathrm{p}}|). \tag{2.9}$$

2.3 Path Integration Dynamics

The neural field provides a mechanism for creating and maintaining a stable activity packet, however as noted in the introduction, the movement of this activity in the absence of external input requires that the packet be asymmetrical. The creation and control of this asymmetry is the focus of PI, where the activity packet can be moved to a arbitrary position in the field without external input. Consider the case of head direction in rodents. It was proposed by Stringer et al. [28] that this movement could be controlled via a set of rotation or movement cells, which would contain synaptic connections to the neural field encoding place or direction. Figure 2.3 illustrates the PI model proposed by Stringer et al.

The movement of the activity packet requires preferential connectivity of clockwise and counterclockwise nodes modulating the collateral connections. This modulation of the synaptic collateral connection strengths cause an asymmetry in the neural field synaptic weights, resulting in a movement of the activity packet [32],[28]. The implementation of these rotation nodes as neurons allow an encoding of the velocity or strength of the movement through the firing rates of the cells, which provide a method an angular velocity to be encoded for head direction, or in two dimensions, a translational velocity for movement in a plane. The incorporation of both the rotational synaptic weights and rates into the neural field dynamics noted above, results in a modification of the field equation (2.3),

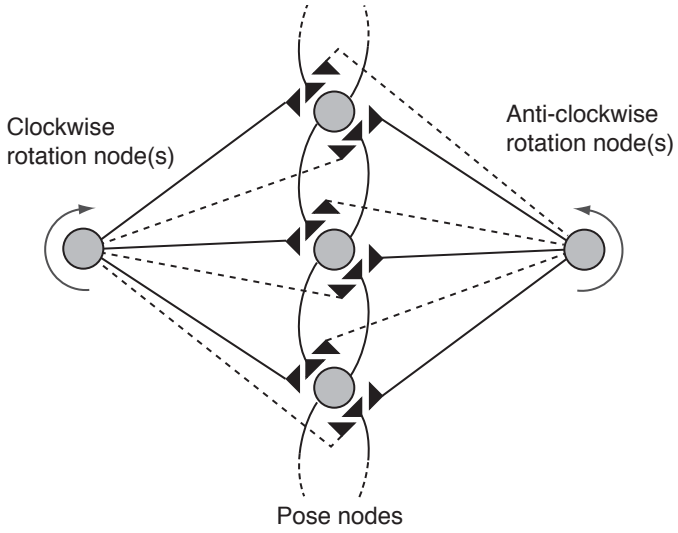


Figure 2.3: The model for path integration, showing a one-dimensional manifold of pose cells with collateral connections (such as cells encoding head directions). The rotation nodes signal rotation velocities proportional to their activity, which is propagated through synapses to modulate the activity of the field, causing an asymmetry in the activation, and moving the activity packet.

$$\tau \frac{\partial h(x,t)}{\partial t} = -h(x,t) + \int_y w^{eff}(x,y,r^{rot}) r(y,t) dy + I^{ext}(x,t). \tag{2.10}$$

Where $w^{eff}(x,y,r^{rot})$ is the combination of the field weights and synaptic influence through the rotational node rates described by $r_i^{rot}(t)$ at time t. The combination of the field weights and synaptic influence of the rotational nodes used by Stringer et al. [28] is described by,

$$w_{ij}^{eff} = w_{ij} \sum_k (1 + w_{ijk}^{rot} r_k^{rot}) \tag{2.11}$$

Earlier work with PI models [38] was centered around the idea of pre-specified weights or transitions, specified during the development of the field. Indeed, some current applications of PI for RatSLAM [12] robotic navigation have implemented a set of pre-specified translations of the activity packet to ensure stable performance of the system over a widely varying set of speeds. In contrast to this method, the work by Stringer et al. [28] showed a method at which these synaptic weights can be learned based on history of movements. This method was based on the notion of a

historical trace integrating a short term memory of related activity in the field during training,

$$1/\eta \frac{\partial \bar{r}_i}{\partial t} = -\bar{r}_i(t) + r_i(t), \tag{2.12}$$

where $\bar{r}$ is the trace of previous activity. The η term controls the amount of influence earlier movements have on the synaptic weights, and can be used to increase the effective asymmetry of the neural field, where a smaller value for η allows for the previous history to weighted heavier, allowing for a larger asymmetry in the field.

The goal of training is to learn an appropriate set of synapses between two or more rotation nodes and the neural field, therefore learning the appropriate asymmetries in the activity packet required for movement. To enable effective learning of the appropriate synaptic weights, the network must associate the firing of a rotation node to a movement of the activity packet in a specific direction and speed [32]. This method is centered around the concept of using Hebbian learning for training the rotational node synapses using the trace term specified in (2.12). Rotation weight training in the PI mechanism is implemented by

$$\delta w_{ijk}^{rot} = \epsilon r_i \bar{r}_j r_k^{rot}, \tag{2.13}$$

where the trace ($\bar{r}$) provides a short term history of the activity packet movement, which is associated with the firing of the rotation cells and the movement of the activity packet, where ϵ is a learning rate.

An interesting effect of the learned PI synaptic weights is an ability of the system to not only learn from a short term history or memory, but also to allow the encoding of asymmetric speeds of movement over specific areas of the field in the movement model itself, rather than just generic motion. The learning of these asymmetries provides the ability for a PI system to learn a specific movement, as illustrated in [29]. This movement could then be reproduced in the exact manner at which they were trained and at differing speeds, depending on the rotation node rates.

2.4 Comparison of Kalman Filters to Path Integration Using Neural Fields

Neural fields, like Kalman filters, also provide an estimate of pose at time t, which is the result of integrating uncertain effects from motor commands to update a model of the previous estimated pose of the system at time $t-1$. One method for comparing the two models is to examine the parameters and function of the models, with respect to the two primary steps of the Kalman filter, the prediction and the correction. This section will provide a novel comparison between these two models, to illustrate where the models are analogous, and where they are different.

As noted above, the prediction step in a Kalman filter is a linear combination of the state at time $t-1$ with a set of motor command results, and uncertainty represented by additive process noise. The result of this is a new pose estimate, $\bar{\mu}_t$. In the neural field, the location of the activity packet is the pose estimate, represented by the node with maximum activity. Although both models provide the a mechanism for state transition, the Kalman filter uses a state transition constant A, whereas the state transition in the neural field is through a learned mechanism using Hebbian learning as specified in Equation (2.7). Motor command results, specified by u_t in the Kalman model, are specified by a rotation rate input $r_i(t)$ into the neural field model. The strength of the motor command result is translated directly to the firing rates in the rotation nodes of the neural field. This input causes an asymmetry in the activity packet, moving the packet to the new location. The movement of the activity in the field is an analogous process to the linear integration of motor command results in the Kalman filter, where the activity packet will be shifted to a new location, forming the uncorrected pose estimate ($\bar{\mu}_t$) at time t. One of the challenges with Kalman filters is correctly modeling the process uncertainty in the system, denoted by R in the algorithm described above. This uncertainty is assumed to be Gaussian, characterized by a mean of 0 and covariance of R. This uncertainty impacts the confidence of the pose estimate through the resulting estimate of the covariance, $\bar{\Sigma}_t$. A major difference between the two models is the lack of a method to

incorporate uncertainty caused by the motor command in the neural model, therefore the integration of the rotation rates causes a direct shift of the estimate.

The resulting pose estimate from the Kalman filter is then refined through the correction step, where an observation is integrated with the expected observation given the results of the prediction step. This step illustrates one of the primary differences to the models, as the neural field does not natively contain a mechanism for updating estimates based on observations from the environment. To provide an effective correction step to the neural field, providing an observation as an external input ($I_{ext}(x,t)$) in (2.10), is proposed, where x is the pose location of the observation. In the Kalman filter, the observation model incorporates uncertainty in the measurement, represented by Gaussian noise, specified by a covariance, Q. The uncertainty can also be represented in the neural field model, where the width of the input pattern $I_{ext}(x,t)$ is specified by the variance in the correction observation. This input creates a new area of activity in the neural field, which influences the existing activity packet through the dynamics of the field. After the external input is provided to the neural field, the competitive dynamics of the model take over. This results in a competition through short distance excitation and long distance inhibition between the existing estimated pose and the external input. The result of this competition is a shifted activity packet, where the corrected pose estimate μ_t is defined by the updated location of maximum activity in the field.

The Kalman filter model combines the distributions of prediction and correction using Bayesian inference [31]. As shown by Wu and Amari [37], neural fields are also an approximate implementation of Bayesian inference, using the combination of the prior estimate with the observation distribution to provide a new distribution, the updated pose. Table 2.1 compares the mechanisms used in the Kalman filter and neural field model for the estimation and correction steps.

By comparing the two models with respect to the parameters illustrates the primary differences in the models listed above. This comparison is illustrated in Table

Table 2.1: Comparison of Kalman filter functional steps with a neural field model using the proposed correction step

	Kalman Filter	**Neural Field using PI**
Prediction	Linear Integration using Motion Model	Path Integration
Correction	Correction using Sensor Model	External Input using updates from (2.10), where uncertainty is specified using width of input

2.2.

Table 2.2: Comparison of Kalman filters parameters with a neural field model using the proposed correction step

	Kalman Filter	**Neural Field using PI**
Motion Uncertainty	Motion covariance parameter 'R'	None
Observation Uncertainty	Observation covariance parameter 'Q'	Uncertainty specified using width of external input
State Transition	Supplied as parameter 'A'	Learned through Hebbian learning
Pose Covariance	Computed using the Kalman gain	Constant

The integration of a correction step in the neural field shows the ability of the field to update an estimate of the neural field based on noisy external observations, such as visual input. In environments where an observation is possible, this allows for a better estimate of pose in the same manner as the Kalman filter. Figure 3.3 illustrates this effect, comparing a Kalman filter which is run using no correction step, integrating the gyro velocities to maintain an estimated pose, with a neural field model integrating the gyro through PI. The pose model can be seen to drift from the actual pose illustrating the need for a correction step based on observations from the environment. Figure 3.4 demonstrates the increase in estimation accuracy using a correction step, where drift is controlled through updates driven by observation.

Although the functionality provided by both models is similar, the differences are illustrated when the parameters and model structure are considered. Although the neural field model does model many aspects of the Kalman filter as seen in Tables 2.1 and 2.2, they contain significant differences in the parameters used in the models. These primary differences are the uncertainty covariance parameters used in the integration of motor command results, the resulting constant pose covariance in the neural field, and the state transition method. These differences, specifically the lack of uncertainty in the motor command results will have an impact on the accuracy of the position which is illustrated using an experimental implementation in Chapter 3. Functionally, the updated PI model using a correction step provides analogous functionality to the Kalman filter, providing updated estimates of pose in consideration of the observation and incorporating uncertainty of the observation into the estimate.

Chapter 3

Using Neural Fields with Path Integration to Implement a Biologically Inspired Gyrocompass

The previous chapter introduced the mechanisms involved with a Kalman filter and a neural field, and compared these models for estimating pose in noisy or sensory deprived environments. Those comparisons further proposed a correction step for the neural field, which would increase the accuracy of the neural field pose model using an observation to correct the estimated pose. This chapter provides an experimental implementation of the above comparison, estimating the pose through the integration of noisy sensors and observations. This experiment will illustrate the performance of the neural field model with a direct comparison to a Kalman filter, and show the relative error between the models. Finally, these models will be compared without a correction step, therefore showing the performance using a simple prediction model based on angular velocity integration.

3.1 Motivation

The previous chapter provided a novel comparison of the common elements and differences between the Kalman filter and neural field models. The goal of this experiment is to demonstrate that comparison, and examine the ability of a neural field with PI in a technological setting where Kalman filters are commonly used, pose estimation. The compared models will be employed to produce corrected and uncorrected pose estimates. The Kalman filter and neural field corrected models will produce pose estimates by integrating a series of angular velocities from a gyro and corrections included from a magnetic compass with added uncertainty in the form of Gaussian

noise. From the perspective of the Kalman filter, this is implementing both the motion step and observation step illustrated in Chapter 2.1. The neural field model will use PI with the updated weight combination method outlined in Chapter 4.2.2 for the motion step, also incorporate the correction updates in Chapter 2.4 to provide the correction step. The uncorrected models integrate the gyro velocities without an observation, which is analogous to maintaining a pose model using idiothetic inputs alone. In this case, both models will maintain a pose estimate using only the linear integration of the results from the motor commands with the previous state, resulting in the motion step of the Kalman filter in Equation (3.1), and the PI without corrections for the neural field. The specific parameters of each model are specified in the Kalman filter and neural field setup sections below. The results of these estimates will be compared to the ground truth, which is the compass headings without added uncertainty through Gaussian noise. The difference in the pose estimates will be used to determine the performance of the models with respect to the error between the actual and estimated pose.

3.1.1 Experimental setup

The experimental setup used a Lego MINDSTORM NXT® system, seen in Figure 3.1, as the primary host for the sensors. Data was recorded from the sensors for off-line processing through the pose estimate models, which were implemented in MATLAB®. The Kalman filter was implemented and verified using the Kalman filter toolbox [15]. The sensors used in this experiment were an NXT® gyro and compass, developed by the Hitechnic® corporation.

The Hitechnic® gyro is a single axis gyroscopic sensor providing +/- 360 degrees per second sensing capability, with a maximum sampling rate of 300Hz. The compass is a lightweight digital compass, providing 360 degree sensing with 1 degree accuracy, and a maximum sampling rate of 100Hz. Both sensors were developed specifically for the Lego NXT®, therefore no specific hardware integration was required, providing 'plug-and-play' functionality. Figure 3.1 shows the NXT® setup used for the data

collection.

Figure 3.1: The Lego MINDSTORM NXT®, configured with Hitechnic gyro and compass sensors

The Hitechnic® compass and gyro were calibrated as per the manufacturer instructions. For the gyro, this involved sampling the gyro on a static surface, to provide an estimate of the bias in the gyro. This value was sampled 600 times to compute an average bias, which was applied to the gyro. In order to calibrate the compass, a turntable was used, which rotated the sensor at 6 degrees per second. This turntable allowed for stable calibration of the sensor, where the compass was rotated through 360 degrees in one direction, then 360 degrees in the opposite direction. At the conclusion of the calibration step, the compass was tested relative to a calibrated compass in a low magnetic environment.

The data recording was implemented using the Python programming language, using the NXT Toolbox [26]. This toolbox is an open source project (GNU GPLv3) available at http://code.google.com. This toolbox provides an interface to the NXT system, including low level connection and communications structures, as well as sensor interfaces. Data was recorded at a rate of 10Hz, and logged as a raw set of (gyro,compass) tuples.

Data was collected by moving the sensor and recording the updated angular velocity and heading information. Uncertainty was added to the compass data using Gaussian distributed noise, characterized by a mean of 0 and a standard deviation of 3 degrees. The resulting heading information provided a set of observations to be used for correction of the filter. Figure 3.2 outlines the raw sampled headings and headings with added uncertainty. The angular velocity and uncertain compass observations were used for the Kalman filter as motor command effects and observations or measurements.

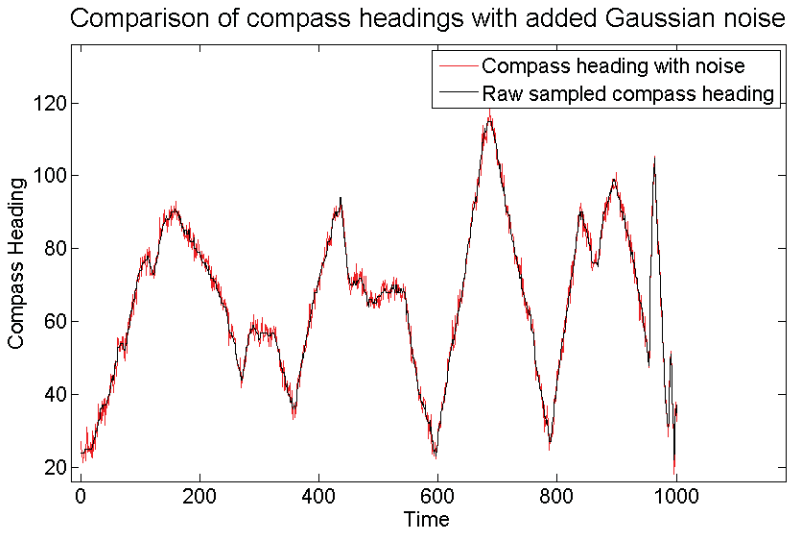


Figure 3.2: A comparison of raw, sampled headings and headings with added Gaussian noise ($\mu = 0$ and $\sigma = 2$). These values show the compass observations to be input into the model compared with the ground truth.

Kalman Filter Setup

For the implementation of the Kalman filter model, the state transition constant A was set to one, allowing for a simple transition based on the effects of the motor command on the previous state. The value of B, the control input constant, was set to the time difference in the sampling rate, denoted as dt. The sampling rate was set to 10Hz, corresponding to $dt = 0.1$. Process uncertainty (R) was fit to a value of 1.5 (degrees) based on the values supplied by the vendor of the sensor. C, the observation constant, was set to 1, as the uncertain pose was directly observed, and

the uncertainty of the observation, Q, was set to a value of 9, corresponding to the Gaussian noise added to the ground truth data to build a set of observations. The gyro inputs ($G(t)$) were mapped to the motor commands (u_t). In consideration of the model equations from (2.1) and (2.2), this results in:

$$\text{Motion model: } x_t = x_{t-1} + dtG(t) + \xi \tag{3.1}$$

and

$$\text{Observation model: } z_t = x_t + \delta \tag{3.2}$$

providing a one dimensional Kalman filter for maintaining a pose estimate.

Neural Field Setup

The Neural field model was developed using the updated weight combination equation in (4.1), and using the proposed observation correction updates presented in Chapter 2 to provide a robust model for PI. This model was developed using 360 nodes, each corresponding to 1 degree. As this model is implemented as a ring, a boundary at 0 and 2π was implemented. The kernel width, controlled by σ_r was set to $\sigma_r = 2\pi/18$, and inhibition was set to $C = 0.5$. Two rotation nodes were implemented, corresponding to clockwise and anti-clockwise motion. For each observation step, the model is updated through the mapping of the uncertain compass observations to the external input to the model from eq. (2.3) ($I^{ext}(t) = O^{compass}(t)$). The standard deviation (σ) which relates the size of the input to the uncertainty of the system was set to 3 to represent the added uncertainty on the observations. Gyro inputs were specified as negative (anti-clockwise) and positive (clockwise), therefore these values were mapped to a clockwise or anti-clockwise node, after being scaled by the sampling rate (dt):

$$r^{rot} = \begin{cases} r^{clockwise} = dtG(t) \text{ if } G(t) > 0. \\ r^{anti-clockwise} = dtG(t) \text{ if } G(t) < 0. \end{cases} \tag{3.3}$$

For trace learning, the window size was set to $\eta = 0.2$, and the learning rate was set to $\alpha = 0.1$. For each simulation, the field was stimulated through external input to develop an activity packet at the pose node corresponding to the first heading. Once this has been formed, the external input was removed and the PI mechanism would be used to control the movement of the packet. This model was developed for the MATLAB® environment, using the 'ode45' differential equation solver to compute activity of the field which is an differential equation solver using the Runge-Kutta algorithm. Furthermore, results were verified using other ode solvers available through MATLAB®. For speed tests of the activity packet, the difference between the previous position and the current position were compared at each time step, taking into account the boundary conditions of the field. To examine the stability of the field, the simulation would increase the rotation rate input over time, causing a ramp up in the activity packet speed during the simulation.

3.2 Results

The data collected was run through both the Kalman and neural field models, in two different configurations. The first configuration was to compare the pose estimate from both models in the absence of an observation for correction. For the Kalman filter this was the prediction step, where the resulting pose estimate was the linear combination of the angular velocities with the initial pose and including uncertainty through additive process noise to produce a new pose estimate. In the case of the neural field, this involved integrating the velocities through the PI rotation rates, and retrieving the pose node which had the maximum activity. Neither model had a correction step applied to update the pose estimate. Figure 3.3 illustrates the comparison results of the Kalman versus the PI model for pose estimates without corrections.

The models were also considered when a correction step was applied, using an observation which was input from the noisy compass model described in figure 3.2. For the Kalman model, this was included in the correction step as an observation,

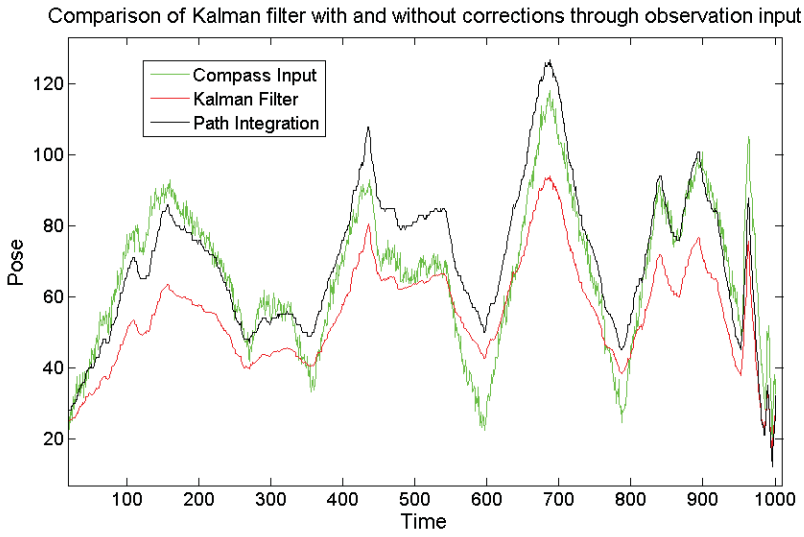


Figure 3.3: A comparison of the Kalman filter versus the neural field model for pose estimation without corrections. This figure demonstrates the accumulation of error or drift when not corrected at each time step.

where Q, the uncertainty covariance, was the same as the variance that was applied to the compass data. The neural field model was updated with the correction step outlined in Chapter 2, where a Gaussian pattern characterized by a mean of the compass observation and a variance the same as the Kalman filter was applied as external input, denoted by I_{ext} in (2.10). Figure 3.4 illustrates the comparison results of the Kalman filter versus the PI model when corrected.

Furthermore, to provide an illustration of the improved accuracy for each model as a consequence of using corrections over a pose prediction only, Figures (3.5(a) and 3.5(b)) are presented. These figures show the results of the model when corrections are applied versus a pose estimate only step, where angular velocities are integrated, but no observations included.

Table 3.1 outlines the difference in error between the models and configurations, where the corrected and uncorrected models are compared to the ground truth of the pose. This ground truth is the compass readings without additive noise, as illustrated in Figure 3.2. For each successive estimate, the error between the actual pose and the estimate pose is collected, then the mean and variance on the error is computed.

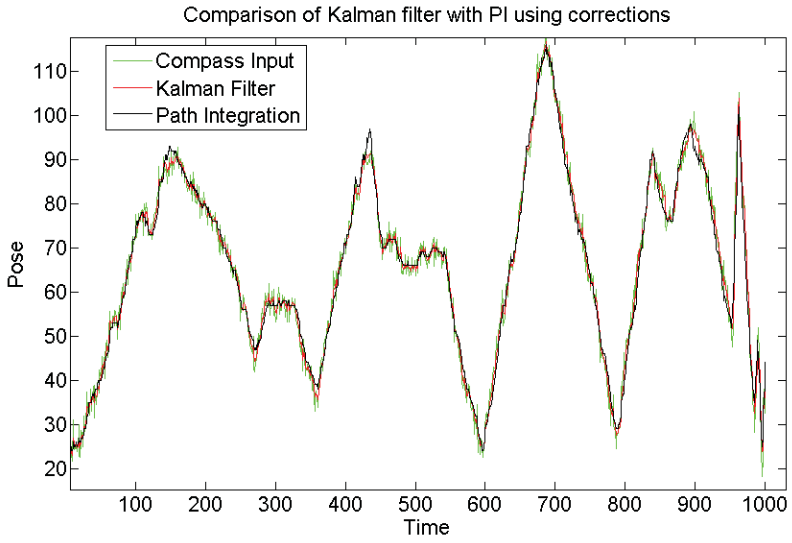


Figure 3.4: A comparison of the Kalman filter versus the neural field model for pose estimation with corrections. This figure demonstrates the lack of accumulation of error or drift when corrected at each time step.

Table 3.1: Comparison of Kalman filters versus neural field model experimental error

	Mean Error (μ)	**Error Variance (σ^2)**
Kalman Filter with Corrections	1.05°	0.67°
Neural Fields with Corrections	1.48°	1.76°
Kalman Filter Pose Estimate	13.08°	62.01°
Neural Field Pose Estimate	8.80°	40.66°

3.3 Discussion

Overall, both models provided a considerable increase in accuracy when the corrections were applied, as shown in Figure 3.5. The integration of the angular velocities suffers from drift, which increases over time due to sampling error, some rounding error in measurement, as the velocities are reported as integers, and jitter in the measurement due to the high sensitivity of the gyro. This compounding drift would continue to grow until corrected, causing a decrease in accuracy which can grow without limit. In the comparison of the models using pose estimate only, or uncorrected

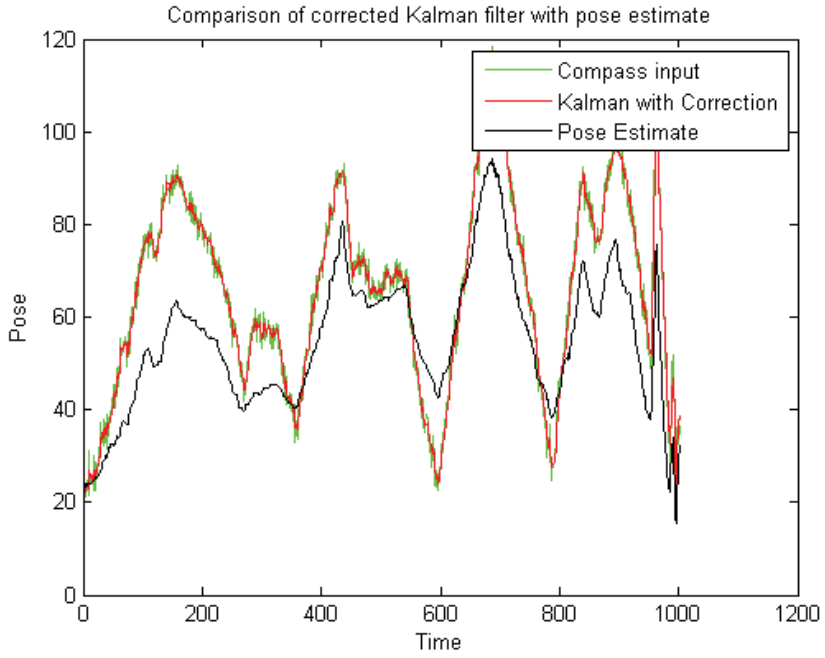
models (Fig. 3.3), this error appears to decrease, however this is due to the periodic nature of the measurement, where the compass was moved through a 100 degree range, changing directions periodically. When a correction step is applied, the accuracy increases considerably over the entire simulation. This increase is caused by the lack of drift error, as the correction step takes place at each time step, minimizing the drift between each successive prediction.

As shown in table 3.1 when corrections are applied, the Kalman filter performs better than the neural field model, showing a low error rate ($\mu = 1.05$ degrees) as compared to the corrected PI model ($\mu = 1.48$ degrees), and a smaller variance on the error of $\sigma^2 = 0.67$ versus $\sigma^2 = 1.76$ respectively. Kalman filters are optimal for linear systems with Gaussian noise, therefore although this result is not unexpected, the corrected neural field model is notable for its performance, approaching the overall accuracy of the Kalman filter model.

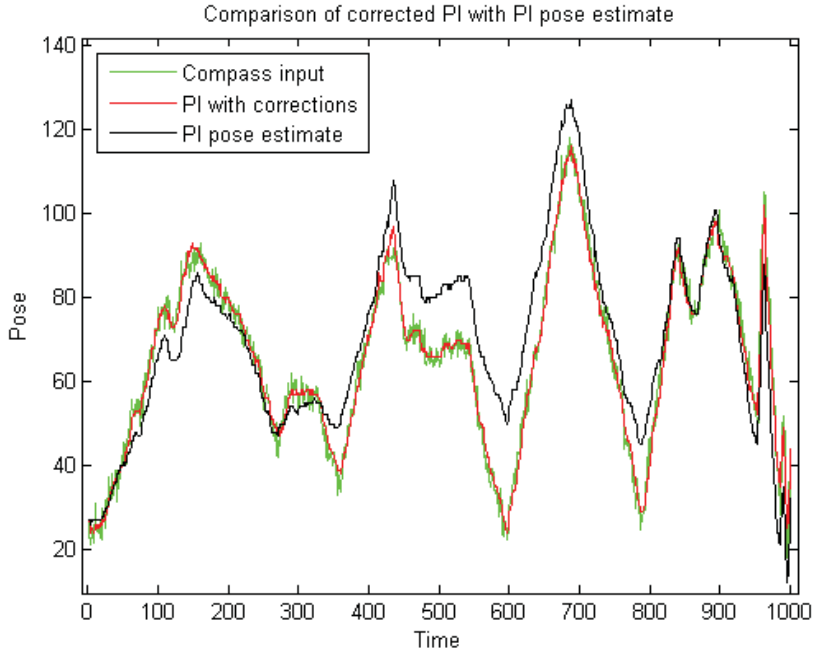
The uncorrected models provide an interesting result, where the neural field model performs slightly better than the Kalman filter model. In this case, the Kalman filter model without corrections suffered more from accumulating drift than the neural field model, where the Kalman filter model had a higher error mean of $\mu = 13.08$ degrees, which a variance of $\sigma^2 = 62.01$ degrees, and the neural field model had an error mean of $\mu = 8.80$ and variance of $\sigma^2 = 40.66$ degrees. This improvement in performance illustrates the 'stiffness' provided in the neural field model caused by the recurrent excitation. The angular velocities for each time step are illustrated in Figure 3.6. For small angular velocities, the recurrent nature of the neural field model results in a resistance to move when input is provided to pose nodes close to the current estimate. The Kalman filter does not have this stiffness however, and will integrate each angular velocity directly. The integration of uncertainty through process noise in the Kalman filter also has an impact on this drift, as it can provide noise which increases, or eliminates a small angular velocity input, whereas the neural field model does not incorporate uncertainty on the motor command results. This difference between the models has little effect on a single sample, however since correction isn't used, these

errors will sum over time to deviate the estimated pose from the actual pose, resulting in the neural field model showing better performance than the Kalman filter.

Overall the results show that for a corrected pose estimate, although the neural field model is not as accurate as the Kalman filter, it shows comparable performance for the model, and provides a biologically plausible implementation of a Kalman filter model, illustrating a method where an accurate pose estimate can be implemented and maintained in the brain.



(a)



(b)

Figure 3.5: Illustration of the effect of using pose estimate correction through observation at each time step. Both the Kalman filter (a) and neural field model (b) showed considerable improvement in accuracy when corrections used, due to the minimization of error at each time step.

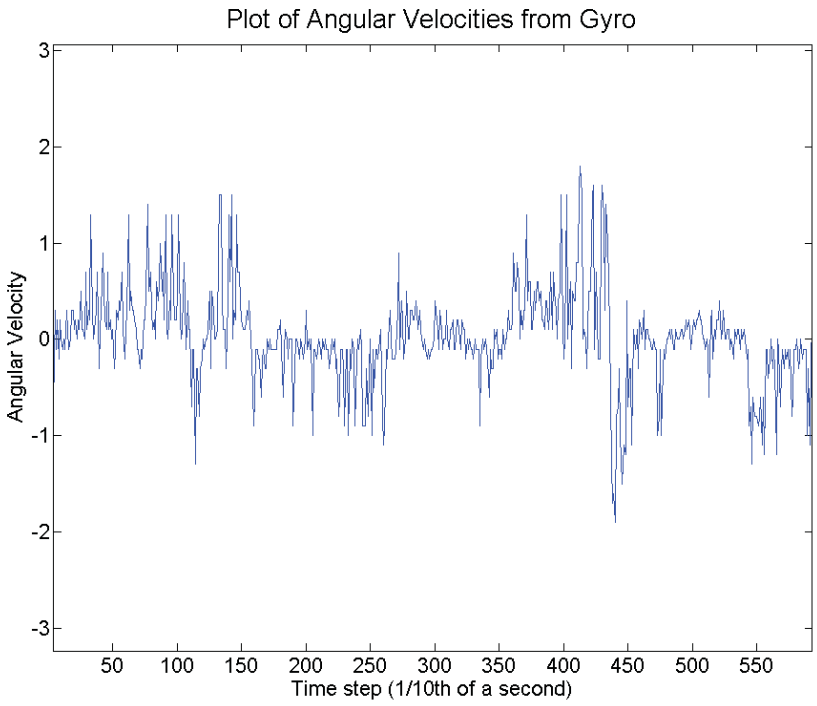


Figure 3.6: A sampling of the angular velocities that were integrated into the models as motor commands at each time step.

Chapter 4

Analysis of Path Integration Robustness

This chapter illustrates the results of an analysis of the mechanisms defined by Stringer et al. [28] for PI. This robustness analysis is with respect to the stability and movement of the activity packet, population decoding methods, and the result of using NMDA stabilization, a method for stabilizing partially trained or noisy synaptic rotation weights. As a result of this analysis, a set of modifications to the model will be proposed, specifically a novel weight combination method to provide a more stable model over a range of rotational inputs. This model will allow for a higher overall speed range available for the PI, stability of the activity packet size, and more stable movement of the packet under strong rotation inputs.

4.1 Results from Robustness Analysis of Path Integration

This section outlines the results of the robustness analysis on the PI mechanism. This section will illustrate potential issues with respect to the PI mechanism which impact the stability of the movement of the activity packet in the neural field. The areas considered in this section include the training mechanism for learning synaptic weights, population decoding mechanisms, and the maintenance of the inhibition in the field, a primary component of the neural field dynamics.

4.1.1 Training

A challenge to neural fields and PI is the creation of perfectly symmetric states to sustain a stable activity packet. As already addressed by Zhang [39], noise in the weights would cause drift of the activity packet thereby deteriorating the pose memory. To provide the perfect symmetry in these weights, the fields should be

trained carefully, considering every node for the same amount of time. This allows for a fully symmetrical set of synaptic weights to be formed. These ideal learning conditions are not always present, which can lead to underdeveloped synaptic weights over parts of the neural field or noisy weight kernels as illustrated in Figure 4.1. The noisy or incomplete training of the field weights can result in asymmetries in the activity packet, causing the packet to drift without rotational input. This effect is illustrated in Figure 4.2, where the activity of the field drifts due to noisy weights without external stimuli or PI.

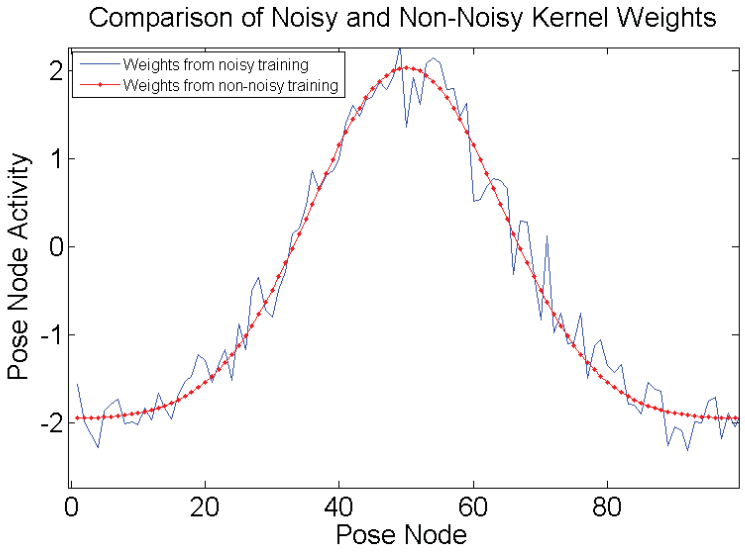


Figure 4.1: Comparison of kernel weights for node 50, showing the typically symmetrical weights that the fields are trained on, and the same weights with noise added.

In the case where incomplete training results in underdeveloped weights, the result is a point attractor network. This network will consist of a series of point attractors, which correspond to the pose nodes with the most developed weights. When an activity packet is formed, it will drift to the nearest pose node with fully developed weights, and then settle over this pose node. Figure 4.3 illustrates the point attractor effect, where a simple neural field of 100 nodes was developed to show the effect of noisy weights. In this plot, a neural field was developed and trained with 10 sets of fully developed weights spaced equally throughout the field. To plot the point attractor effect, 100 simulations were run, one for each node, where input was provided

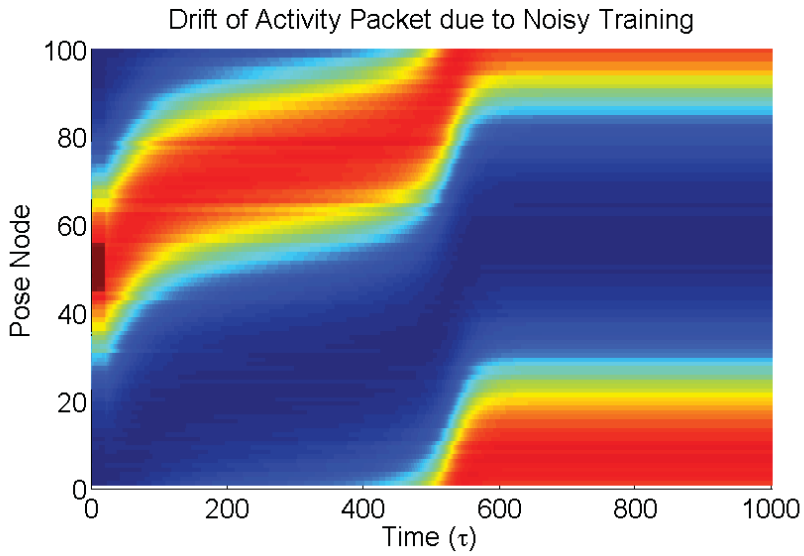


Figure 4.2: Surface plot of the field activity for a simple, 100 node neural field when noisy weight kernels are used. The asymmetries caused by the noisy weight kernels causes drift in the activity packet, resulting in inconsistent activity in the field.

into each pose node of the field and the movement of the activity packet recorded. The composite plot in Figure 4.3 was developed over time to show the point attractor effect, where each trace in the plot is the movement of an activity packet formed over each pose node. These traces show the activity packets formed over each node converging to the closest point attractor.

As noted above, to provide a stable neural field for PI, a stable, non-drifting activity packet and therefore symmetrical neural field weights are required. Symmetrical weights can be developed through careful training where each node is considered with the same inputs and for the same time. If this is not possible, then a stabilization mechanism such as the NMDA stabilization which is considered in this work can be used to counteract the effect of asymmetrical or noisy weights.

4.1.2 Population Decoding

Population decoding is the process of decoding information which has been encoded in the firing rates of a neural field. For this thesis, the variable that is decoded from the field is the head direction, which is represented by the location of the activity packet.

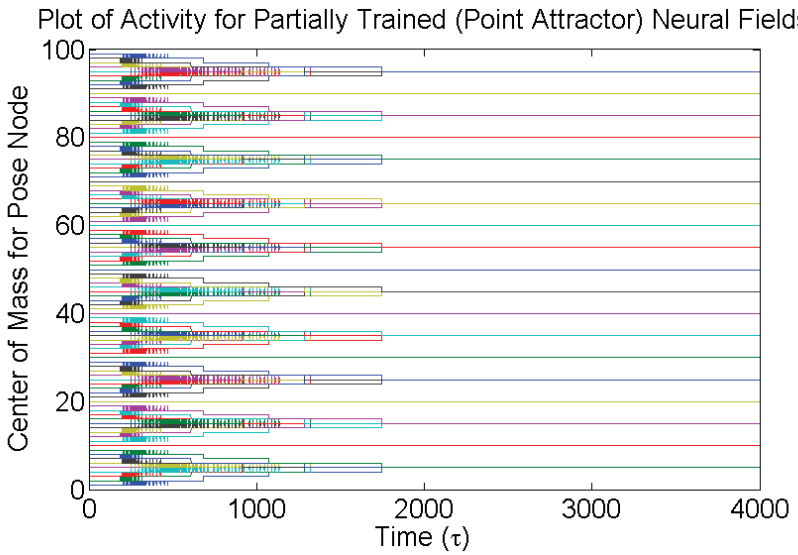


Figure 4.3: The centers of mass for activity packets formed from partial training, resulting in a point attractor type field, where only 10 node weight kernels were fully developed

Ideally, this value could be decoded by examining the node which corresponds to the maximum activity, however this assumes a perfectly symmetrical activity packet, where the node that has the maximal activity is the center of the packet. This assumption poses a challenge with respect to noisy or underdeveloped weights and fluctuations in the shape of the activity packet. In cases of noisy or underdeveloped weights, such as those in Figure 4.1, the noise may cause nodes which are not the center of the activity packet to be the node with maximal activity, resulting in an incorrect decoding of the head direction.

The second challenge is with respect to fluctuations in the shape of the activity packet. The method of trace learning can illustrate where max activity decoding can cause fluctuations in the packet shape, and return an incorrect result. As noted in chapter 2, the trace method has the desirable characteristic that the rotational weights are learned from the previous motion of the activity packet, therefore not requiring any pre-specified weights for transitions. Although this method provides a memory-based learning method, it also can cause a slight inconsistency in the training of the weights, due to a lack of initialization of the trace term. As the trace term starts from a silent state (zero), a condition arises under strong input where the earlier

nodes in the training cycle have smaller values than the weights later in the training cycle. Figure 4.4(b) illustrates this effect, where the magnitude of weights trained first are lower than subsequent weights. During PI, this can cause a change in shape of the packet, as the difference in weight amplitudes affect the recurrent input to the activity packet. This change in shape can influence the decoding considerably. As the packet moves over this area of underdeveloped weights, the decoding will select the maximum activity, which will not be the center of the packet. When the change in activity over time is used to calculate a speed of the activity packet, this incorrect decoding will result in inconsistent speeds being calculated. This behaviour can be seen in Figure4.4(a), where incorrect population decoding resulted in an apparent spiking in the speed of the activity packet.

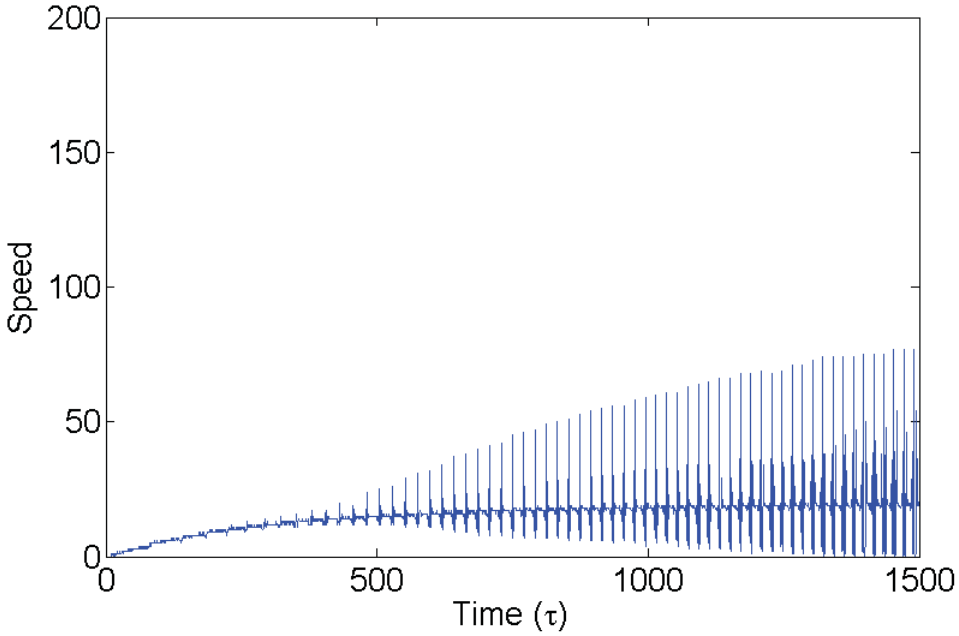
To correctly decode the activity of the neural field, one method is to consider the center of mass of the activity packet. This center of mass would be equivalent to the max activity in the case of a symmetrical activity packet, however it is also insensitive to distortions in the activity packet due to underdeveloped weights or asymmetries introduced into the packet through movement.

The center of mass method used for population decoding in this work is the normalized weighted sum of the activity, defined by,

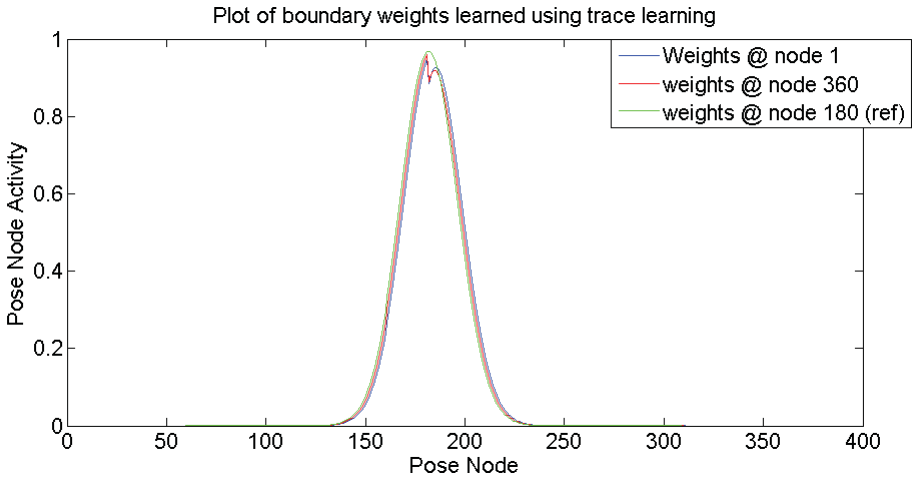
$$CoM = \sum_i r_i i \Delta x \tag{4.1}$$

where i is the index of the pose node, and r_i is the activity at node i. This poses a challenge with the boundary cases, as the activity can span the boundaries. In this case, the activity packet is translated to a location where boundary conditions are not required and the center of mass is computed. This center of mass location is then translated back to the actual position of the activity packet.

The result of this population decoding algorithm is an insensitivity of the decoding mechanism to the shape of the activity packet. Figure 4.5 illustrates the robustness in the algorithm for decoding noisy or underdeveloped weights. This figure was generated from the simulation used in Figure 4.4(a), using center of mass decoding rather



(a)



(b)

Figure 4.4: Effects on the rotation activity (a) and weight matrix (b) at 0 by training using an initially empty trace rule. The pulsing effect in (a) is caused by using max activity decoding when the activity packet moves over the inconsistent weights in (b), which has been shifted over node 180 for comparison. The weights at node 180 are provided as a reference of properly formed weights. This figure shows the weight differences on the boundary between node 360 and 1.

than the location of the maximum activity. The pulsing effects are not visible, as the center of mass decoding provides a stable change in position during PI, resulting in

a stable speed calculation.

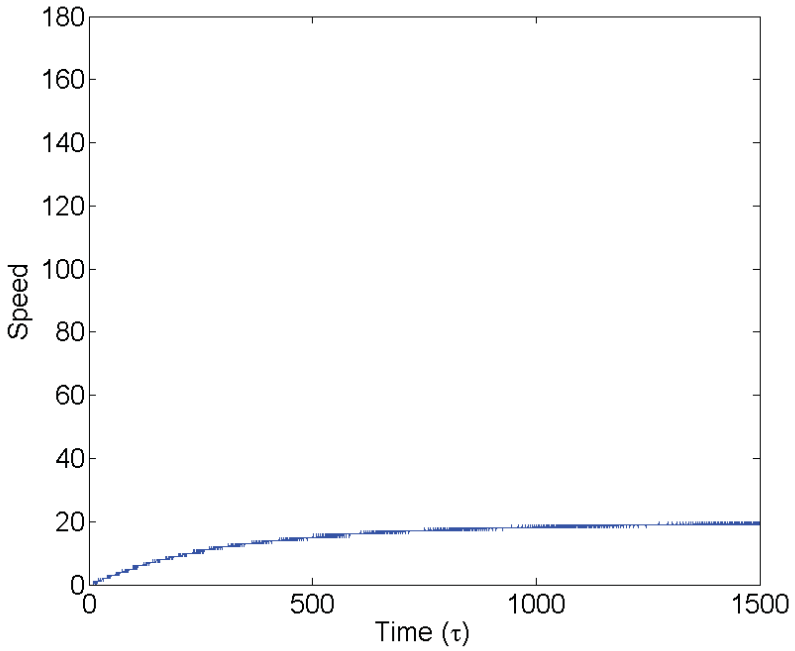


Figure 4.5: The simulation shown in Figure 4.4(a) using center of mass population decoding, which is insensitive to changes in the shape of the activity packet.

4.1.3 Maintaining Appropriate Inhibition During Weight Combination

The inhibition of the neural field is critical to ensure that a stable activity packet is formed. As each node is fully connected to all other nodes, distance dependent excitatory input is provided to all nodes when a particular pose node is excited. The inhibition in the field counteracts this effect, resulting in a single, localized packet of activity. In the case where the inhibition is insufficient or absent, this results in the uncontrolled growth of the activity packet, and a decrease in the utility of the field as it would become fully excited [1]. A careful maintenance of inhibition is key to both the neural field and PI, to prevent an unstable state of the network where the activity packet grows uncontrolled. This effect is shown in Figure 4.6, where the rotation input is increased over time. The increase in the activity packet size results in the field eventually breaking down, which is shown by the high activity (in red) of the entire field beyond $t = 700$. The key to any weight combination method for

PI is to combine the weights in a meaningful way that allows for an asymmetry in the activity packet, while maintaining the inhibition at a rate where the packet size remains stable.

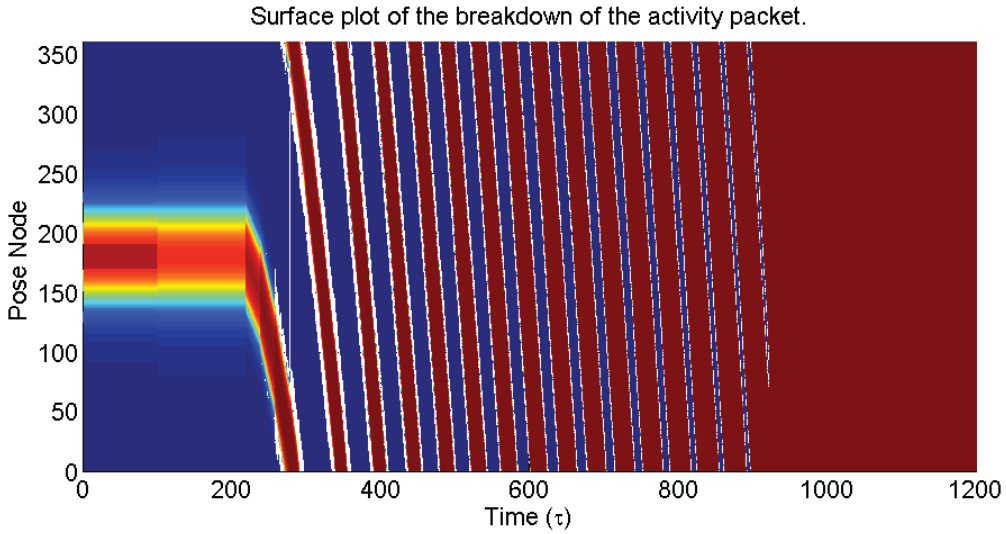


Figure 4.6: Surface plot illustrating PI using rotation rates that are ramped up over time. This plot shows the growth of the activity packet, and eventual breakdown of the field under strong rotational input.

The existing method for weight combinations in Equation 9 uses the addition of a constant to allow for the resulting effective weights to maintain inhibition, therefore maintaining some stability. This addition of a constant is required as the rotation weights have a minimum value of zero (Fig. 4.7), therefore do not contain inhibition. The multiplication of the neural field weights which include inhibition and PI weights which do not would result in an effective weight kernel which does not contain any inhibition. This loss of inhibition results in the field moving to a stable state of full activation. The addition of the constant to the rotation weights allows the inhibition to be maintained during the multiplication of the weights, however it also has the undesirable effect of increasing the amplitude of the weight kernel. This increase in amplitude increases the amplitude of the neural field kernel, which will lead to activity packet growing due to the limit of the sigmoid activation function and eventually breaking down. Any alternative method for weight combination must seek to minimize the growth of the activity packet, while maintaining the inhibition to ensure

that the packet remains stable.

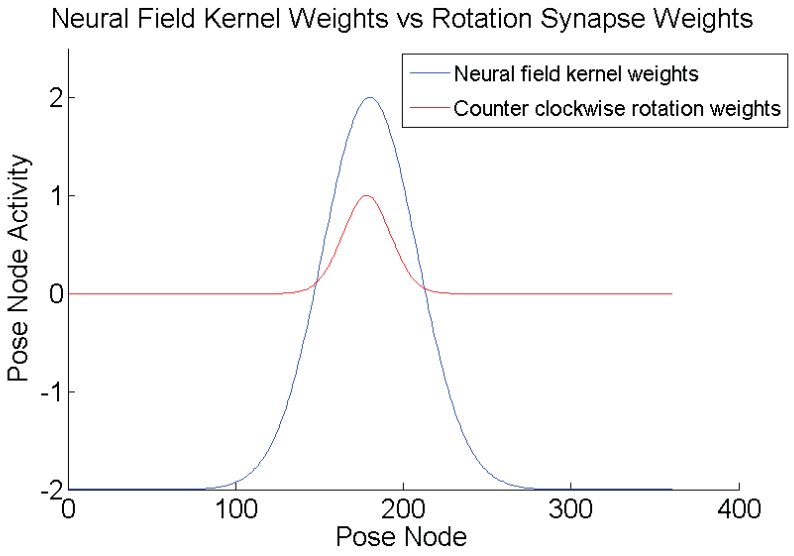


Figure 4.7: Comparison of neural field kernel and rotation synapse weights. The rotational synaptic weights, which have a minimum value of 0, require care when combining multiplicatively with the kernel weights, to ensure that the inhibition is not removed from the field. Without this inhibition, the weights would result in a growing state where the entire field becomes excited.

As noted above, the combination of weights and rotation rates for PI are multiplicative, therefore they run the risk of increasing in amplitude to orders of magnitude larger than the neural field weights. This increased amplitude due to rotational input causes a growth in the activity packet width, which if uncontrolled, can result in an inability of the field to maintain a localized solution of pose as the maximal activity is over a large area. Figure 4.8 illustrates this, where under a low rotational input, there is a noticeable increase in the activity packet width. This effect can also cause an unstable growth in the network during PI, resulting in a growing packet which eventually saturates the field. Figure 4.9 illustrates this effect, where rotation rates are increased to illustrate the growth of the activity packet.

To provide control of the activity packet size during PI, a method to control the amplitude of the effective weight kernel is required. One possible technique would be to apply a normalization to stabilize the amplitude of these weights, when used after the rotation weights and rates are combined with the neural field. Using the existing

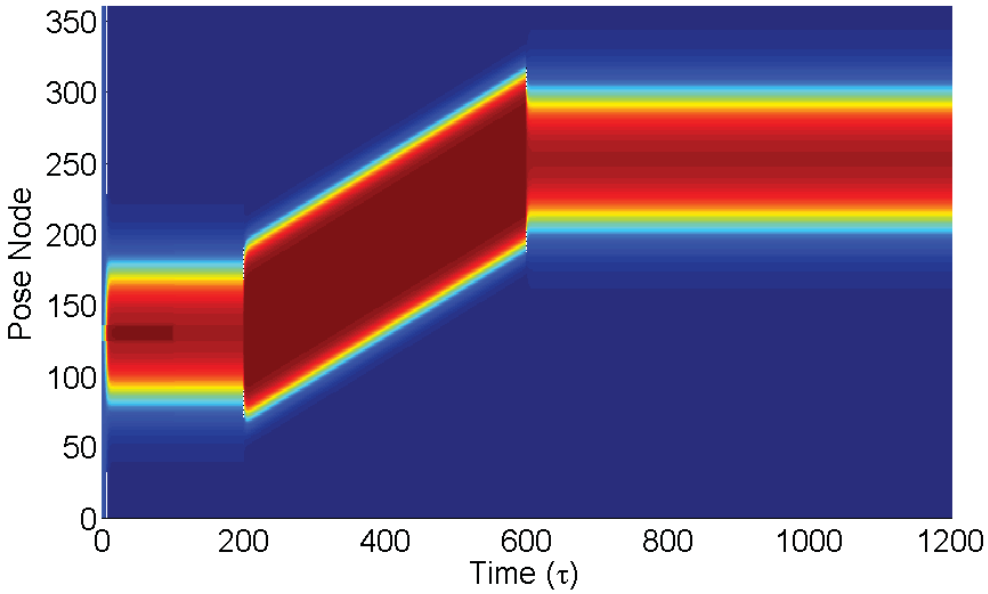


Figure 4.8: Illustration of the activity packet using PI with a low rotational input between $t = 200$ and $t = 600$, showing that even under low rotational input, the activity packet will increase in width, due to the high activation rates.

weight combination method however, the normalization does not have the desired effect. Consider the combination of the rotation rate and synaptic weights (Fig. 4.7). The PI rotation weights have no inhibition (negative component), therefore a positive constant (1) is added to the result of the weights multiplied by the rotation rates in Equation 9 to maintain the fields inhibition during the weight combination. This has the effect of amplifying the positive weights further, while leaving the inhibition at the same rate. When the scaled rotation weights are combined with the field's weights through multiplication, the inhibition is maintained at the fields rate, however the excitation (positive) portion of the activation is scaled up. A simple normalization of these resulting activations to a range that is equivalent to the field's rates, will shrink the peak, but it will also decrease the inhibition, effectively shifting up the kernel. As the rotation strength increases, eventually the inhibition is decreased to approach zero. The decreased inhibition causes growth of the activity packet, eventually causing the field to be fully excited and break down.

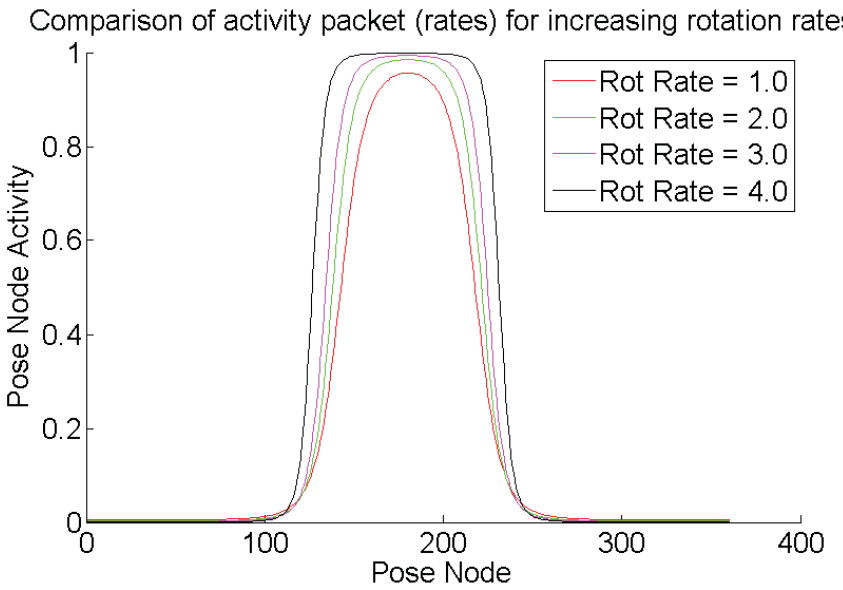


Figure 4.9: Comparison of Activity Packet size under differing rotation rates, showing the effective growth of the activity packet under strong rotational input. This growth eventually leads to an uncontrolled excitation of the field.

4.2 Proposed Path Integration Modifications

This section proposes methods to address the issues noted in the previous analysis of the PI mechanism. This section proposes a new weight combination method to provide a higher control on the size of the activity packet, which results in a higher overall stability and speed of the packet under strong rotational input, while maintaining inhibition at an appropriate rate to support the dynamics of the neural field. Finally, the method of NMDA stabilization is considered to stabilize the field when noisy or inconsistent learning has taken place, allowing for stable movement of the activity packet and dampening the drift of the packet due to noisy synaptic weights as illustrated in Figure 4.3.

4.2.1 Alternative Weight Combination Methods

The current weight combination method, as outlined in Equation 2.11, introduces specific characteristics to the PI mechanism which can be limiting in application.

The goal of this work is to examine alternative weight combination methods to provide a more robust implementation of the PI method which provides more stability in the kernel weights, while increasing the overall range of speeds of the PI with respect to rotational input. Specifically, it is desirable to find a method which controls the growth of the activity packet, limits any instabilities caused through learning, maintains the required inhibition in the field, and increases the overall speed range of the rotational input.

In this section we propose a new method of combining the rotation weights and kernel weights, to eliminate or greatly limit the effects noted above, while maintaining or improving the effectiveness and computational tractability of PI. The weight combination mechanism listed in (2.11) uses a multiplicative combination of the synaptic rotation weights, the firing rate of the rotation node, and the collateral weights of the neural field. As noted in the previous analysis (4.1.3), this combination causes a very rapid growth of the activity in the field, and amplifies any inconsistencies in the rotational weights caused by the trace learning.

Trace learning results in a set of synaptic weights that are slightly skewed from the overall kernel weights in the neural field. This slight distortion, when combined with the neural field weights, result in a skew of the weight kernel in the appropriate direction. As the goals of the weight kernel is to cause an asymmetry in the activity packet, rules which increase this skew will increase not only the overall top speed at which the activity packet will move, but also the sensitivity of the asymmetry to input from the rotational nodes.

The proposed method is a modification of (2.11) to use the addition of the neural field kernel weights with a shifted version of the rotational weights trained from (2.13), and scaled by the rotation speed. This method is described by

$$w_{ij}^{eff} = w_{ij} + \sum_{k} (w_{ijk}^{rot} - \mu) r_k^{rot} \tag{4.2}$$

Where w_{ij}^{eff} is the effective neural field weights from i to j, w_{ij} is the neural field kernel weights, w_{ijk}^{rot} is the trained rotational synaptic weights, and r_k^{rot} is the rotational

synaptic strength. Any value of the inhibition constant ,μ, can be subtracted from the weight kernel, as long as this value maintains the overall inhibition of the field at an appropriate rate. We use in the following work the mean of the rotational synaptic weight. This method removes the multiplication of the neural field and rotational weights, while still maintaining inhibition in the resulting effective weight kernel, through the subtraction of the mean value for the kernel. The training of both the neural field and the rotational synaptic weights are identical, and normalization was applied to the weights after training.

4.2.2 Offsetting Sensor Drift Using NMDA Stabilization

To counteract the stability issues noted with noisy and partial training in neural fields, a non-linear activation method has been applied to the activity of the pose cells in a similar manner to Stringer et al. [28]. This proposal considered that the non-linear activation could be implemented biologically through the effects of voltage dependent ion channels, such as NMDA receptors. When neurons are at rest, these receptors are blocked by magnesium ions, and are inactive. An increase in the membrane potential of the neuron will remove this block, allowing the neuron to fire easier. Using a non-linear activation such as NMDA receptors results in a group of neurons which can be activated with lower stimulus at time $t+1$ if they were active at time t. This method can be implemented in the model through the variation of a offset on the sigmoid activation function, depending on the field activity in the previous time step. The offset for the sigmoid activation function in (2.4) is described by,

$$\alpha_i = \begin{cases} \alpha^{high} & \text{if } r_i < \gamma. \\ \alpha^{low} & \text{if } r_i \geq \gamma. \end{cases}$$

Where γ is a threshold, and α_i is the sigmoid offset. This method has the effect of dampening the activation of the nodes in the field which were not firing above the threshold in the time step before, and therefore strengthens the recurrent input to the nodes that are active. It can be considered a boost of inhibition for the field, with the exception of the activated nodes. Path integration requires a stable set of field

weights, therefore the qualities of NMDA stabilization are attractive in cases of noisy field weights or irregular training.

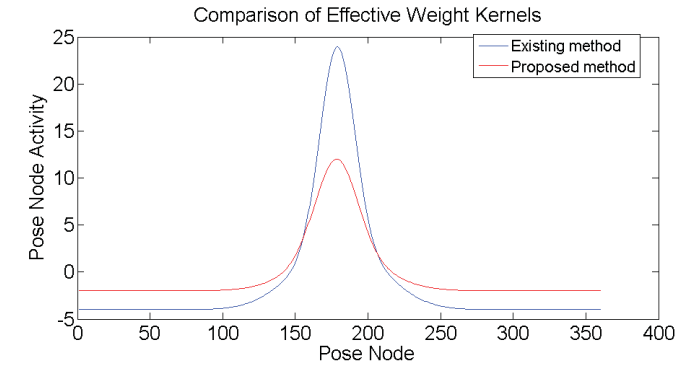
4.3 Results of Proposed Path Integration Mechanism Modifications

The following section outlines the effects of the proposed PI mechanisms with respect to the robustness of the neural field. The modifications proposed in the previous section are examined, and results presented to show that the use of the proposed updates provide a stable PI mechanism that provides stability under partial or noisy training, as well as high rotational input, while maintaining the inhibition in the field to allow for stable dynamics.

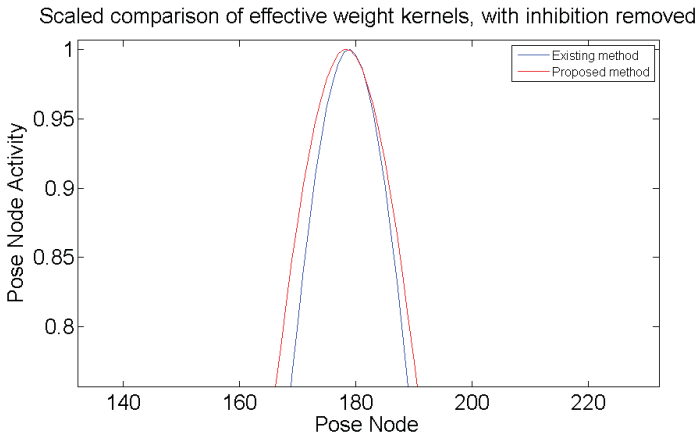
4.3.1 Weight Combination

Through addition rather than multiplication of the weights, the overall asymmetry of the kernel is increased, which is illustrated in Figure 4.10. The addition supports the widening of the kernel, specifically on the appropriate side of the required rotation. This allows more recurrent input to distort the activity packet, thereby moving the activity in a more efficient manner than the multiplicative method. The existing weight combination method (2.11) amplifies the distortion at high weight values (e.g. near the center of mass), but maintains a narrow profile for the weights (Fig 4.10)(a). The modification of the effective weight kernel allows for a wider range of the activity packet speeds, which results in a higher sensitivity of the PI for a given set of rotation strengths. The addition further has the effect of increasing the amplitude of the activity at a much slower rate as seen in Fig. 4.10 which, when run through a sigmoidal activation function such as (2.4), results in a narrower activity packet. This narrower activity extends the range of rotational input that can be maintained by the field before an explosive growth of the activity packet.

The movement speed of the activity packet is controlled through the asymmetries in the effective weight kernel. This speed, which can be increased by increasing the asymmetry, has an asymptotic speed limit, and can be noted in Figures 4.4 and 4.11.



(a)

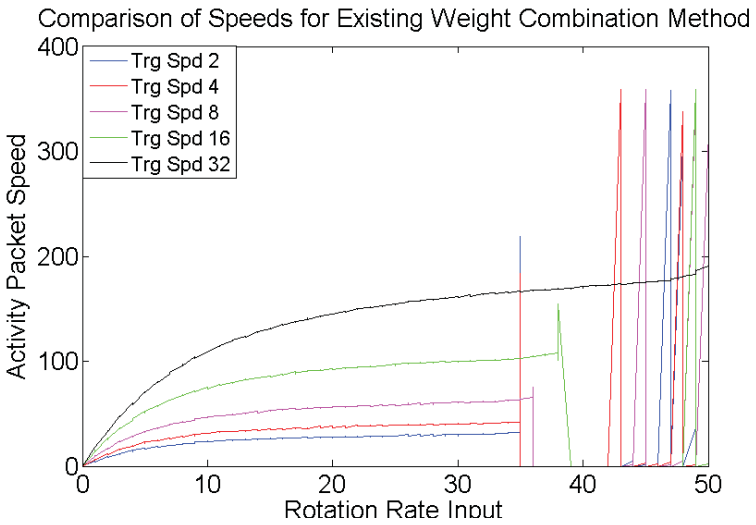


(b)

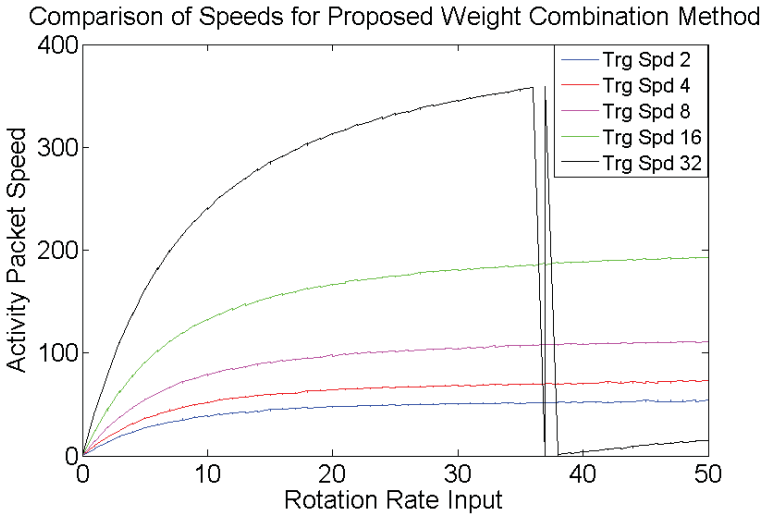
Figure 4.10: Comparison of effective weight kernels of node 180 for anti-clockwise rotation of a 1-D field. Plotted together (a), the amplitude difference between methods is shown, where the existing method grows rapidly, resulting in rapid growth of the activity packet leading the field to be fully excited. When scaled together and with inhibition removed, the differences in the kernel weights become more apparent, where the proposed method contains more skew in the counter clockwise direction (left), allowing more recurrent input in the activity packet, creating the desired asymmetry. This is illustrated in (b), where a portion of the weights have been magnified to view the weight differences.

As noted above, the proposed method for weight combination results in a wider, more asymmetrical input to the field, thereby allowing for a higher maximum speed as well as stability under a higher range of inputs from the rotational nodes as shown in Figure 4.11(b).

Figure 4.11 compares the movement speeds of both the existing weight combination method (a), and the proposed new weight combination method (b). These



(a)



(b)

Figure 4.11: Comparison of existing weight combination method (a) and the proposed weight combination method (b).This comparison illustrates the higher top speed insensitivity to the spiking behaviour, higher overall speed, and stability under a larger range of rotational inputs.

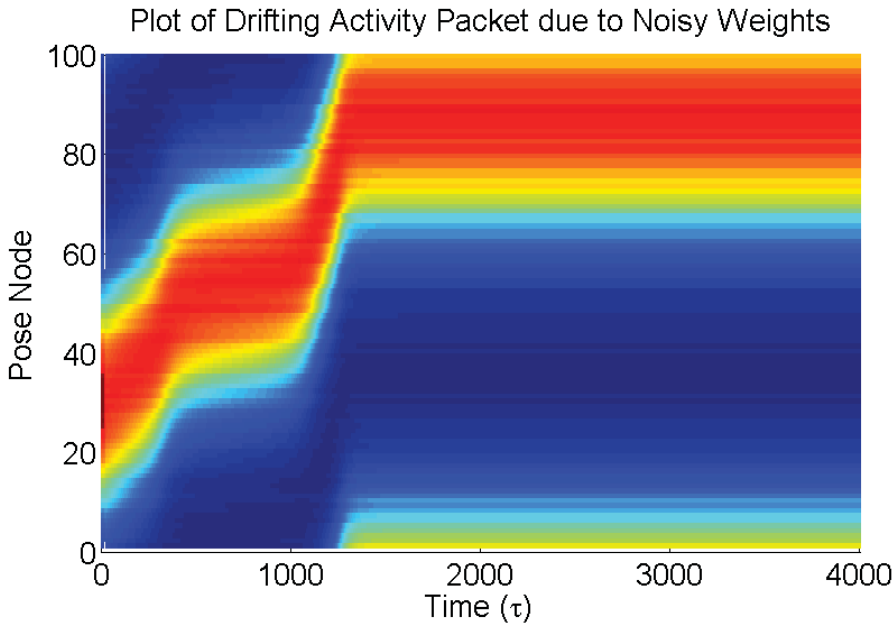
methods were compared by increasing rotational input to the fields on kernels trained using different training speeds. The training speeds are defined by the rate of movement that is shown to the trace during training, therefore allowing the trace to learn that a given movement and rotation rate input covers a larger area of the field. The goal of this figure is to illustrate the difference in speeds across different training

rates, as well as the stability of the field to high rotational input. As illustrated in Figure 4.11(b), the proposed method achieves a higher overall speed of more than two times the existing method. Furthermore, the proposed method shows a higher speed range for the same set of rotational inputs, allowing for more robust control of the activity packet speed. Through the additional rather than multiplicative combination of the synaptic weights with the field weights, the proposed method shows a higher resistance to the breakdown of the field due to growth of the activity packet, resulting in stability over higher rotational inputs. This effect is illustrated in Figure 4.11, where the field breaks down under increasing rotational input (a). This breakdown in the field is prevented in (b), showing increased stability over (a). The improvements from the proposed weight combination method can be noted in Figure 4.11, where the speed of the activity packet for fast training (32) shows the exceeding the size of the field at rotation rate input greater than 37. This high speed results in a wrapping effect in the figure, while still showing stable activity.

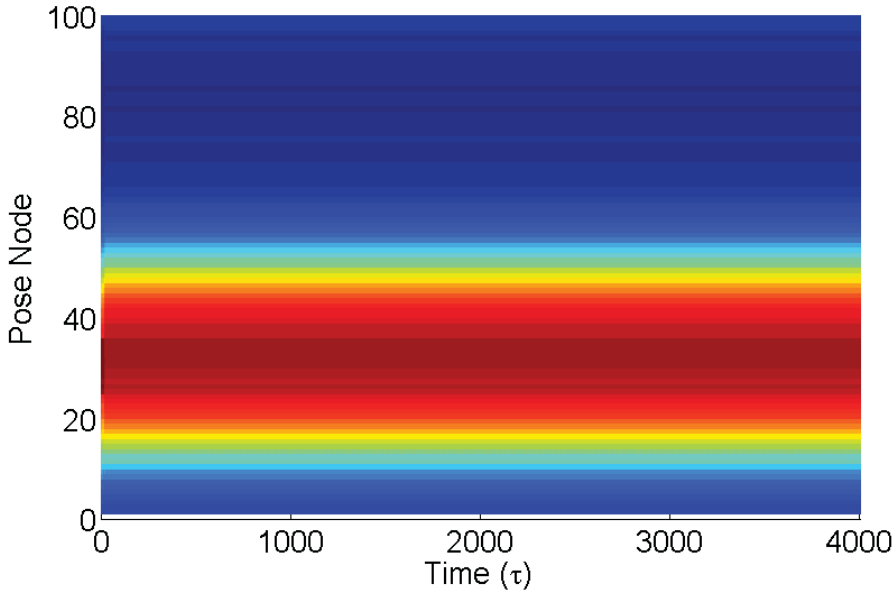
4.3.2 NMDA Stabilization of Path Integration

This section will demonstrate that NMDA stabilization is an effective stabilization technique for PI. Intuitively, the non-linear activation using thresholds allows for an added level of resistance to movement or 'stiffness' in the activity packet. This resistance is defined by the sigmoid thresholds α^{high} and α^{low}. In the case of a noisy kernel, which results in an activity packet that drifts, this stabilization results in a fully stable activity packet. Figure 4.12 illustrates this stabilization effect, showing the activity when trained with noisy input, causing drift in the packet (a). The NMDA stabilization dampens the noise sufficiently to maintain a stable activity packet, as shown in (b).

As noted in Section 4.1.1, partially developed weight kernels can result in a point attractor network, as illustrated in Fig. 4.3. For partially developed weight kernels, a high value of α_{high} reverses the point attractor behaviour of the field, however this value creates a strong 'stiffness' to the activity packet, requiring strong input at



(a)



(b)

Figure 4.12: Comparison of the effects of non-linear activation through the application of a firing rate dependent threshold. Figure (a) shows the asymmetrical activity packet which resulted from noisy weights in 4.1, and (b), the result of using non-linear activation to dampen the drifting through boosting the most active neurons

alternate locations of the field to form a new activity packet. Figure 4.13 shows the effect of NMDA stabilization using a value of $\alpha_{high} = 10$ for the excited nodes on the

partially trained field in Figure 4.3 which resulted in a point attractor. Using NMDA stabilization results in a reversal of the point attractor behaviour, allowing for stable activity across the field.

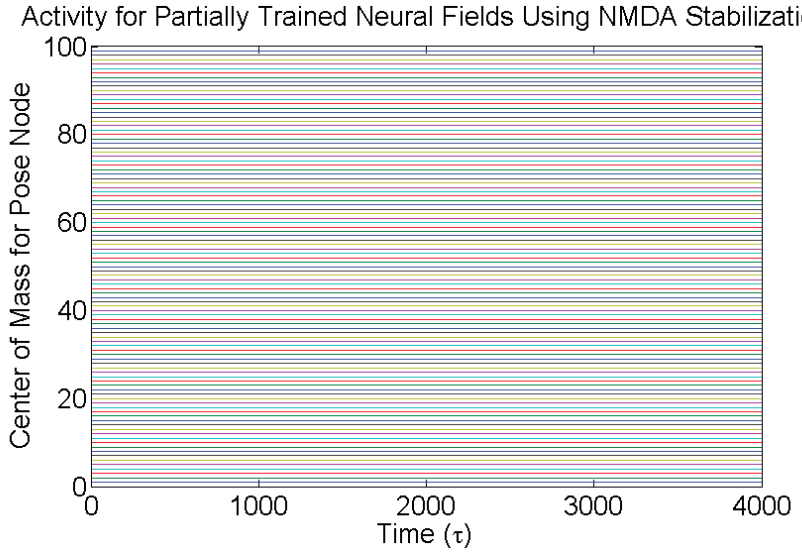
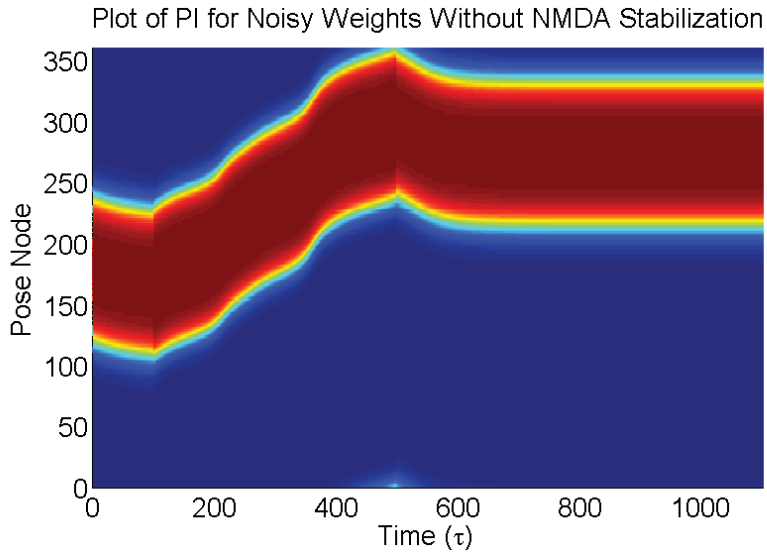


Figure 4.13: Comparison of the effects of non-linear activation through the application of an firing rate dependent threshold on the partially trained network shown in Fig. 4.3. The result of using non-linear activation with a high α_{high} value (10) to dampen the point attractor effect through boosting the most active neurons, which effectively counteracts the point attractor effect.

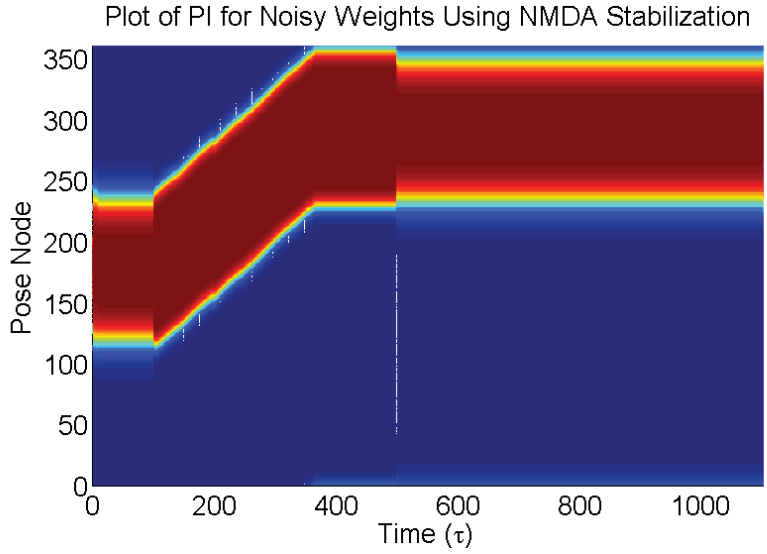
In consideration of PI, NMDA stabilization is effective at stabilizing the field during movement, as well as when the rotation nodes are not firing. This allows for a stabilization of the activity packet from drift when not moving, but also smoother movement of the activity packet when trained on noisy weights. Figure 4.14 illustrates the smoothing effect. The field was trained on noisy weights, causing both drift in the activity packet when at rest, but also inconsistent movement of the activity packet when rotational input is applied (Figure 4.14(a)). In the figure, rotational input was applied from $x = 100$ to $x = 500$, and moving the packet with noisy weights. When the NMDA stabilization was applied, the packet was stabilized both when rotational input was applied, and also when the rotational nodes were at rest (Figure 4.14(b)).

This form of stabilization requires an increase in the rotation input to control

the activity packet. This is a side effect of the resistance of the activity packet to move due to stabilization. This increased input is offset by the utility of the stabilized packet, as it shows stability from drift and consistent movement of the activity packet when a rotational input is supplied.



(a)



(b)

Figure 4.14: Comparison of the effects of non-linear activation through the application of an firing rate dependent threshold on a field trained with noisy input which is used for path integration (a) a high α_{high} value (10) to dampen noise effects. The asymmetries introduced through the noise in the weight training results when no rotational input is supplied (t=0-100, t=500-1100) and inconsistent movement when PI is applied (t=100-500). The application of NMDA (b) dampens the effect of drift, and allows for stable field activity both with and without rotation input

Chapter 5

Conclusion

The goal of the preceding work is to present a biologically plausible model for integrating pose estimate using idiothetic inputs, and correcting that pose estimate through observations from the environment. This work was decomposed into two parts, the first being a comparison of the neural field model to a commonly accepted, optimal model for linear state transitions with uncertainty represented by Gaussian noise - the Kalman filter. This thesis has presented both models, and compared them functionally to draw conclusions on the capabilities of both for creating a pose estimate, and correcting that estimate through observation. The goal of the second part was to examine the neural field model with the PI mechanism to propose updates to the model for robustness. The results of both sections are illustrated in an experiment to produce a pose estimate, integrating motor command updates in the form of angular velocities from a gyro which is corrected by a noisy compass observation.

The Kalman filter provides a strong baseline for comparison, as these filters are commonly used for pose estimation in many fields successfully. The question arises of how animals can integrate inputs to provide an estimate of pose, which is necessary for navigation in the environment. In consideration of pose estimation in animals, a commonly applied model for maintaining pose is the neural field model, as outlined in Chapter 1.

The Kalman filter can be decomposed into two major steps, the prediction step and the correction step. This work proposed analogous functionality in the neural field model, showing how an estimate can be produced and corrected in a biologically plausible method. For prediction, the Kalman filter completes a linear combination of the previous pose with motor commands and additive noise to produce a new pose estimate. The neural field model uses PI, the application of asymmetric input to the

activity packet through a set of synaptic weights modulated by a set of rotation nodes to provide an analogous function. As the activity packet is the pose estimate in a neural field, the movement of this activity due to motor commands via the rotation node results in the development of a new pose estimate. This functionality is common with the Kalman filter, however the neural field model does not incorporate uncertainty in the motor commands through process noise, therefore it uses the rotation input directly to translate the activity packet. The second step, correction, allows for the integration of an observation to update the pose estimate. In the Kalman filter, these are integrated through innovation, or the difference between what is expected to be observed at the pose estimate, and what was actually observed, with some additive Gaussian noise which models uncertainty on the measurement. This work showed that an analogous correction can also take place with the neural field where a new area of activity localized over the node corresponding to the observation can be introduced to the field. The the measurement noise in this observation is captured with the width of the correction input. The result in both cases is a corrected pose, which is then used as the basis for the next pose estimate.

Experimentally, the results of these steps showed the performance of both the Kalman filter and neural field model in both predictive (Fig. 3.3) and corrected (Fig. 3.4) pose estimates. To produce a noisy set of motor commands and observations, a gyro and compass were used to collect angular velocities and headings respectively. These, along with additive Gaussian noise, were input into both models and the results presented in Chapter 3. As expected, the Kalman filter performed best, showing a mean error of 1.05 degrees, with a variance of 0.6 degrees, showing a very accurate prediction. The neural field however performed nearly as well, with a mean error of 1.48 degrees, with a variance of 1.76 degrees. When corrections were removed, both models suffered, however the neural field model performed better than the Kalman model. This is explained by the resistance of the neural field to small acceleration values due to recurrent input, as well as the lack of uncertainty modeled by process noise in the system, as compared to the Kalman model. The interesting difference

in performance is with respect to the corrected versus uncorrected models. In both cases, the corrected models did considerably better than the uncorrected model, even though the observations were uncertain. The cause of this difference in performance is the accumulation of drift due to the lack of correction. Both of the prediction (uncorrected) models use a linear combination of the previous state with motor commands to produce a new state. Each state prediction has an element of drift over time, and this drift is accumulated to produce worse estimates without the correction steps. Although the correction steps will not completely remove all error, they limit the drift in each integration step, therefore limiting the overall drift of the pose estimate over time.

The second major component of this thesis is the result of an analysis of the existing PI method proposed by Stringer et al. [29], and considers issues of performance and stability. To allow for a neural field model to integrate both idiothetic and observational inputs for an estimate of pose, the model must be robust under a wide range or rotational inputs and in consideration of noisy or incomplete training. This thesis explored the boundaries of performance with respect to the PI mechanism, specifically related to training speed and rotational input. In consideration of the behaviour noted in the analysis, most aspects were traced back to the way that the synaptic weights from the rotation nodes and the field weights were combined. Issues discovered in chapter 4 included the effects of noisy or partial training, the rapid growth of the activity packet, and the low speed range of the field with respect to rotational inputs.

Partial or noisy training for the PI synaptic weights has a large effect in the movement and stability of the activity packet, therefore causing instability of the motion and the field. These instabilities can result in a drifting kernel when no rotational input is supplied to the field, and also inconsistent behaviour of the activity packet during PI. The application of a non-linear activation method such as the NMDA stabilization defined in Chapter 3 for the field results in a stabilization of the rates in the field, which in turn results in not only a removal of drift in the activity

packet, but also stable movement of the activation when rotational input is applied. Although this method has the desirable quality of resisting noise and incomplete training, it also will limit the competition effect of the field. The selection of the threshold, as well as the offsets α^{high} and α^{low} is critical to ensure that these values resist drift, while allowing for the formation of a new activity packet at a different area of the field. Using PI with NMDA stabilization requires higher rotational input due to the stiffness of the field, however as PI involves the combination of weight kernels between the field and rotation weights, this combination causes a smoothing effect for noisy weights. Due to this effect the value of α^{high} could be lowered, therefore lowering the required rotation input for the packet movement.

The rapid growth of the activity packet was caused by the method at which the weights were being combined. The multiplication of the weights caused a rapid increase in the amplitude or intensity of the excitatory weights, however it did not conversely increase skewness or asymmetry of the packet. This behaviour results in a rapid increase in the activity of the field, but not a proportional rapid increase in the speed of the bubble movement through skew in the weight asymmetry, or sensitivity to rotational input. This mechanism, combined with the sigmoid activation function, resulted in the rapid width increase in the resulting activity packet, thereby increasing the collateral excitatory input given to other nodes in the field. As this excitatory input increased, the field was driven to an overall higher state of activity, eventually resulting to the entire field being excited and breaking down. Although it is ideal to consider normalization to correct this, the rapid amplitude increases resulted in normalization decreasing the inhibition, unfortunately causing the very effect the normalization was attempting to prevent.

The proposed weight combination rule allows for a more robust implementation of the weight combination mechanism for PI. This robustness is shown in a higher overall top speed, a greater stable speed range for control of the activity packet through rotation rates, and a more robust control of the activity packet size. Computationally, this method reduces a complex convolution of weights to a simple addition,

which increases the computational tractability of the rule when extended to multiple dimensions.

For modeling human or animal behaviour, this method provides a technique for better control of the activity packet size, and therefore stability of the simulations. For non biological applications (e.g. cognitive robotics), this method allows for a faster movement of the packet based on input, and a more consistent movement of the packet. A robust implementation of the PI model allows for a stable implementation of a pose model under a wider range of rotational inputs. Extended to two dimensions, PI allows for navigation, however it requires stable control of the field for integration of low level accelerations. The proposed kernel not only offers a stable solution to the PI models proposed for animals, but could also be applied to other model tasks where a high stability and better control is required.

Overall, this work illustrated that a modified neural field model with a PI mechanism for updating the activity packet location is an effective, biologically plausible method for predicting and correcting pose estimates in robotics. Computationally, this model is less efficient than the Kalman filter, as the dynamic equations must be updated each time step, whereas the Kalman is considerably more tractable. With this in mind however, the goal of this thesis was not to propose an alternate method for a Kalman filter, but to examine a biologically plausible method for how animals maintain pose estimates. Furthermore, the model accounts for the modification of those pose estimates through idiothetic inputs and update those estimates through external input, for example a visual input. The updated neural field model results in a robust, biologically inspired model using Bayesian inference to effectively predict a pose. This model provides analogous functionality to a Kalman filter, in effect functioning as a biologically plausible Kalman filter.

Bibliography

[1] S. I. Amari. Dynamics of pattern formation in lateral-inhibition type neural fields. *Biological Cybernetics*, 27:77–87, 1977.

[2] R. S. Bucy and P. D. Joseph. *Filtering for stochastic processes with applications to guidance*. Interscience Publishers New York, 1968.

[3] W. Erlhagen and E. Bicho. The dynamic neural field approach to cognitive robotics. *Journal of Neural Engineering*, 3:36–54, 2006.

[4] D. Jancke, W. Erlhagen, G. Schoner, and H. Dinse. Shorter latencies for motion trajectories than for flashes in population responses of a cat primary visual cortex. *Journal of Physiology*, 556.3:971–982, 2004.

[5] J. S. Johnson, J. P. Spencer, S. J. Luck, and G. Schoner. A dynamic neural field model of visual working memory and change detection. *Psychological Science*, 20(5):568–77, 2009.

[6] R. E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D):35–45, 1960.

[7] C. Kaminski, T. Crees, J. Ferguson, A. Forrest, J. Williams, D. Hopkin, and G. Heard. 12 days under ice; an historic auv deployment in the canadian high arctic. In *Autonomous Underwater Vehicles (AUV), 2010 IEEE/OES*, pages 1 –11, sept. 2010.

[8] A. Kelly. A 3D State Space Formulation of a Navigation Kalman Filter for Autonomous Vehicles. Technical Report CMU-RI-TR-94-19, Robotics Institute, Pittsburgh, PA, May 1994.

[9] J. C. Kinsey, R. M. Eustice, and L. L. Whitcomb. A survey of underwater vehicle navigation: Recent advances and new challenges. In *IFAC Conference of Manoeuvering and Control of Marine Craft*, Lisbon, Portugal, September 2006. Invited paper.

[10] J. J. Leonard, A. A. Bennett, C. M. Smith, H. Jacob, and S. Feder. Autonomous underwater vehicle navigation. In *MIT Marine Robotics Laboratory Technical Memorandum*, 1998.

[11] B. L. McNaughton, F. P. Battaglia, O. Jensen, E. I. Moser, and M. Moser. Path integration and the neural basis of the 'cognitive map'. *Nature Reviews Neuroscience*, 7:663–678, 2006.

[12] M. J. Milford, J. Wiles, and G. F. Wyeth. Solving navigational uncertainty using grid cells on robots. *PLOS Computational Biology*, 6, 2010.

[13] M. J. Milford, G. F. Wyeth, and D. Prasser. Ratslam: A hippocampal model for simultaneous localization and mapping. In *Proceedings of IEEE International Conference on Robotics and Automation*, pages 403–408, 2004.

[14] P. A. Miller, J. A. Farrell, Z. Yuanyuan, and V. Djapic. Autonomous underwater vehicle navigation. *Oceanic Engineering, IEEE Journal of*, 35(3):663 –678, july 2010.

[15] K. Murphy. Kalman filter toolbox. http://www.cs.ubc.ca//~murphyk/Software/Kalman/kalma 2000–2004. University of British Columbia.

[16] K. Paliwal and A. Basu. A speech enhancement method based on kalman filtering. In *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP '87.*, volume 12, pages 177 – 180, apr 1987.

[17] N. Pelavas, G. J. Heard, and C. E. Lucas. Autonomous underwater vehicle localization using the acoustic tracking system. *J Acoust Soc Am*, 132(3):2054, 2012.

[18] A. N. Ramjattana and P. A. Cross. A kalman filter model for an integrated land vehicle navigation system. *Journal of Navigation*, 48:Allison N. Ramjattan and Paul A. Cross (1995). A Kalman Filter Model for an Integrated Land Vehicle Navigation System. Journal of Navigation, 48, pp 293–302 doi:10.1017/S0373463300012765, 1995.

[19] E. T. Rolls, S. M. Stringer, and T. P. Trappenberg. A unified model of spatial and episodic memory. In *Proceedings of the Royal Society*, volume 269, pages 1087–1093, 2002.

[20] S. I. Roumeliotis, G. S. Sukhatme, and G. A. Bekey. Circumventing dynamic modeling: evaluation of the error-state kalman filter applied to mobile robot localization. In *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on*, volume 2, pages 1656 –1663 vol.2, 1999.

[21] A. Samsonovich and B. L. McNaughton. Path integration and cognitive mapping in a continuous attractor neural network model. *The Journal of Neuroscience*, 17:5900–5920, 1997.

[22] G. Schoner. Dynamical systems approach to cognition. In R. Sun, editor, *The Cambridge Handbook of Computational Psychology*. Cambridge University Press, Cambridge, 2008.

[23] G. Schoner and C. Engels. Dynamic field architecture for autonomous systems. In *From Perception to Action Conference, 1994, Proceedings*, pages 242 – 253, sept. 1994.

[24] H. W. Sorenson. Least-squares estimation: from gauss to kalman. *IEEE Spectrum*, 7:63–68, July 1970.

[25] H.W. Sorenson. On the development of practical nonlinear filters. *Information Sciences*, 7(0):253 – 270, 1974.

[26] Various Contributors Open Source. Nxt-python. http://code.google.com/p/nxt-python/, 2009–. Supported by GoogleCode, GNU GPL3.

[27] S. M. Stringer, E. T. Rolls, T. P. Trappenberg, and I. E. T. de Araujo. Self-organising continuous attractor networks and path integration: two-dimensional models of place cells. *Computation in Neural Systems*, 13:429–446, 2002.

[28] S. M. Stringer, E. T. Rolls, T. P. Trappenberg, and I. E. T. de Araujo. Self-organizing continuous attractor networks and path integration: one-dimensional models of head direction cells. *Network: Computation in Neural Systems*, 13:217–242, 2002.

[29] S. M. Stringer, E. T. Rolls, T. P. Trappenberg, and I. E. T. de Araujo. Self-organizing continuous attractor networks and motor function. *Neural Networks*, 16:161–182, 2003.

[30] J. G. Taylor. Neural 'bubble' dynamics in two dimensions: foundations. *Biological Cybernetics*, 80:393–409, 2000.

[31] S. Thrun, W. Brugard, and D. Fox. *Probabilistic Robotics*. MIT Press, 2006.

[32] T. P. Trappenberg. *Fundamentals of Computational Neuroscience*. Oxford University Press, Oxford, 2002.

[33] T. P. Trappenberg, M. Dorris, D. P. Munoz, and R. M. Klein. A model of saccade initiation based on the competitive integration of exogenous and endogenous signals in the superior colliculus. *Journal of Cognitive Neuroscience*, 13:256–271, 2001.

[34] J. Wang and W. J. Wilson. 3d relative position and orientation estimation using kalman filter for robot control. In *Robotics and Automation, 1992. Proceedings., 1992 IEEE International Conference on*, pages 2638 –2645 vol.3, may 1992.

[35] H. R. Wilson and J. D. Cowan. Excitatory and inhibitory interactions in localized populations of model neurons. *Biophysics Journal*, 12:1–24, 1972.

[36] H. R. Wilson and J. D. Cowan. A mathematical theory of the functional dynamics of cortical and thalamic nervous tissue. *Kybernetik*, 13:55–80, 1973.

[37] S. Wu and S. I. Amari. Neural implementation of bayesian inference in population codes. In *NIPS*, pages 317–323, 2001.

[38] X. Xie, R. H. R. Hahnloser, and H. S. Seung. Double-ring network modeling of the head direction system. *Physics Review*, E66, 2002.

[39] K. Zhang. Representation of spatial orientation by the intrinsic dynamics of the head direction cell ensemble: A theory. *Journal of Neuroscience*, 16:2112–2126, 1996.

[40] S. K. Zibner, C. Faubel, I. Iossifidis, and G. Schoner. Dynamic neural fields as building blocks of a cortex-inspired architecture for robotic scene representation. *IEEE T. Autonomous Mental Development*, 3:74–91, 2011.

Appendix A

Copyright Release Request

February 1, 2013

Journal of Cognitive Computation (Copyright held by Defence Research and Development Canada)
9 Grove Street, Dartmouth NS

I am preparing my Master of Computer Science thesis for submission to the Faculty of Graduate Studies at Dalhousie University, Halifax, Nova Scotia, Canada. I am seeking your permission to include a manuscript version of the following paper(s) as a chapter in the thesis:

Improved Path Integration Using a Modified Weight Combination Method, Warren Connors, Thomas Trappenberg, Journal of Cognitive Computation, 2013

Canadian graduate theses are reproduced by the Library and Archives of Canada (formerly National Library of Canada) through a non-exclusive, world-wide license to reproduce, loan, distribute, or sell theses. I am also seeking your permission for the material described above to be reproduced and distributed by the LAC(NLC). Further details about the LAC(NLC) thesis program are available on the LAC(NLC) website (www.nlc-bnc.ca).

Full publication details and a copy of this permission letter will be included in the thesis.

Yours sincerely,

Warren A. Connors

Permission is granted for:

a) the inclusion of the material described above in your thesis.

b) for the material described above to be included in the copy of your thesis that is sent to the Library and Archives of Canada (formerly National Library of Canada) for reproduction and distribution.

Name: _____ Title: _____

Signature: _____ Date: _____