a) Sensor Applications for Exploring Dynamical Systems with Max/MSP, Part I: An Interactive Toolbox

b) Ashlae Blum

c) ABSTRACT

To deepen the understanding of a complex parameter space, it is highly useful to develop an analytical intuition with respect to the state variables of a dynamical system. Especially when working with real-world data, we stress the importance of using a natural-feeling way to navigate through the state space. Informed by this motivation, we aim to explore cases of complex dynamical systems where state equations and parameters may be highly multidimensional, that is, having multiple parameters which may vary simultaneously across the system. It will therefore be quite handy to have a fast and efficient way of manipulating multiple parameter changes at the same time. While this can be done in a limited fashion on a screen with a mouse and keyboard, there is certainly something left to be desired from this method. If we are able to move the controls of such parameter spaces into a 3D environment, the creative and analytical possibilities as a research tool immediately unfold.

The goal of this project is to implement an innovative method for viewing and interacting with complex dynamical systems. We have designed a real-time interactive system for using motion sensors to navigate complex systems by developing a toolbox of models for these systems. Using the LSM6DS3 inertial measurement units and the Arduino UNO, a hardware setup is interfaced with the XBee S2C using serial communication. Sensor data from this setup is then sent to the Max/MSP graphical programming environment, which holds an interactive model of a complex dynamical system based on the Kuramoto model for complex networks [1]. Beginning with models of the inverse square law and damped driven harmonic oscillators, these systems will be developed into a multi-oscillator network, where parameter variations of coupling, phase, and frequency will determine the evolution of the system. In the Max environment, our sensor controls are designed to allow real-time navigation through multiple parameters at the same time. This allows for a quick, natural, and intuitive way of manipulating multidimensional state variables. The graphical outupts are shown as particle systems, chosen for their physical modeling capabilities [6].

d, e) THE PROBLEM & ITS SIGNIFICANCE

The analysis of complex dynamical systems poses many challenges, including: 1) the ability to fluidly navigate parameter spaces to see the eventual outcome of system behaviour, 2) developing an intuitive understanding of what the equations represent, and 3) dealing with multidimensionality and causal relationships between these parametric dimensions [9]. Since many dynamical systems are modeled using data from real-world systems, they should have extensible ways of being explained in terms of physical phenomena. As such, it seems useful to build upon and develop a language framework for communicating about these systems. In the development of programming languages, for example, the use of everyday syntax and grammar has allowed a linguistic universality which is easily understood, scalable, and applicable all over the world. We create a collection of simple models as tools, recognizing that complicated systems are often approximated as corrections added to solvable systems [18]. Following this line of thought, if a gestural language can be built to explore such types of data-inclusive research spaces, interesting extensions of data analysis may be achieved. The approach used in this project incorporates ideas from several areas of physics, engineering, and mathematics. Included are components from nonlinear dynamics, chaos, complex networks, and remote sensing. After a brief introduction

to these subjects, we will connect the dots by detailing the methodology used in this project, results obtained, and conclusions.

f) EXISTING THEORETICAL APPROACHES

A conceptual example: the Lorenz attractor.
With such a broad spectrum of research topics as found in complex dynamical systems, we turn first to the classic case of the Lorenz attractor. The Lorenz attractor is the solution to the Lorenz system of ordinary differential equations, which exhibits chaotic dynamical behavior, and is highly sensitive to changes in initial conditions as well as parameter state fluctuations [10]. It is an instructive example for analysis of complex systems, because its properties are interesting and have clear, continuous geometrical representations. Through analysis of its phase space, system behaviour and topological features such as convergence, divergence, and regions of attraction are determined. These also include instantaneous state variables, stability conditions, and types or rates of stability. Using visual and graphical computing approaches, short- and long-term behaviours can be analyzed to surmise interesting behaviours which emerge across the system [8].

Practical extensions: the Kuramoto Model and Complex Networks.
Extending said concepts, an innovative approach for modeling compex networks can be found in examples from physics. The nodes of a complex network can be modeled by treating them (and the system at large) as a set of coupled of harmonic oscillators. The Kuramoto model for synchronization can be used to describe this scenario as a large set of weakly-coupled, phase-dependent oscillators,

$$\dot{\theta}_i = \omega_i + \frac{\lambda}{N}\sum_{j=1}^{N}\sin(\theta_j - \theta_i), i = 1 \ldots N \quad ,$$

where N is the number of oscillators in the system, each having phase θ, and a coupling strength λ [1]. Solving the continuity equations leads to several distinct parameter spaces which exhibit diverse state transition behaviours as λ is varied through its range [2]. The Kuramoto model has been widely used to analyze behaviour of complex networks with varying degrees of regularity and randomness, such as small-world networks and scale-free networks [11], [12]. Real-world examples can be seen in modeling complex power grid networks [14], [3], and international finance [17].

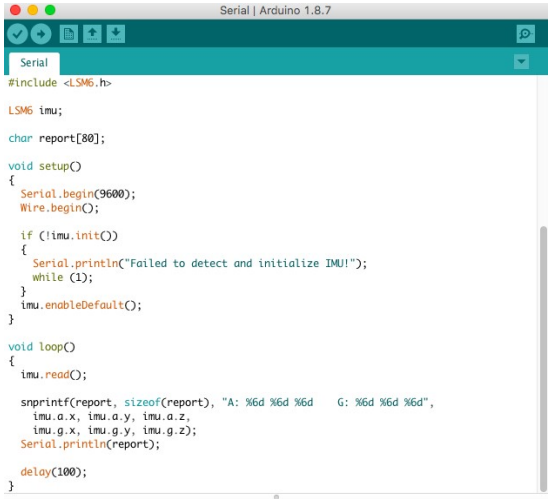
One notable feature used in the characterization of scale free networks is the power law, given by $P(k) = k^{-\gamma}$, where for $k \epsilon$ [1, ∞) the distribution has a unique, well-defined mean only if $k > 2$, and has finite variance only if $k > 3$ [5]. This distribution curve is referred to in statistics as a long-tailed distribution, where the total value of some quantity above certain values of k, defined as the area under the curve of $P(k)$, continues to be appreciably large even as k increases far from the central distribution [5]. In nature, most systems follow power laws with well-defined means, but not variances, such that behaviour outside of the above limits of defined parameter k does not have a unique solution.

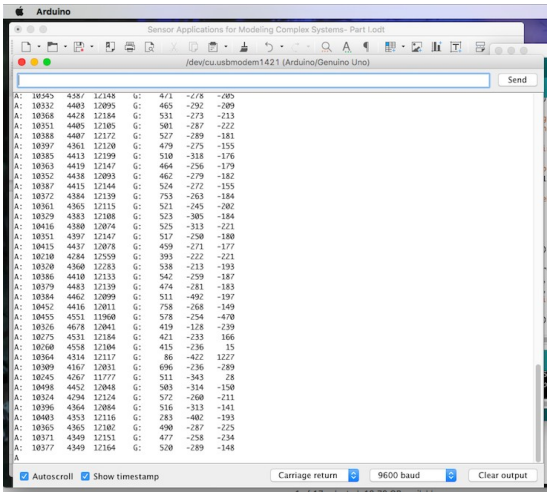
As discussed above, the modeling of systems through complex dynamics can illustrate a variety of topological properties. This information can tell us how state variables will evolve as parameters are continuously varied. The behaviour that a system experiences along the way, from small perturbation to large perturbations, can give qualitative as well as quantitative information about what happenes in different regions of the parameter space [9]. Information like this can be used not only to model real systems, but to design systems which have specific types of behaviour under specific constraint conditions [13].

g) METHODOLOGY

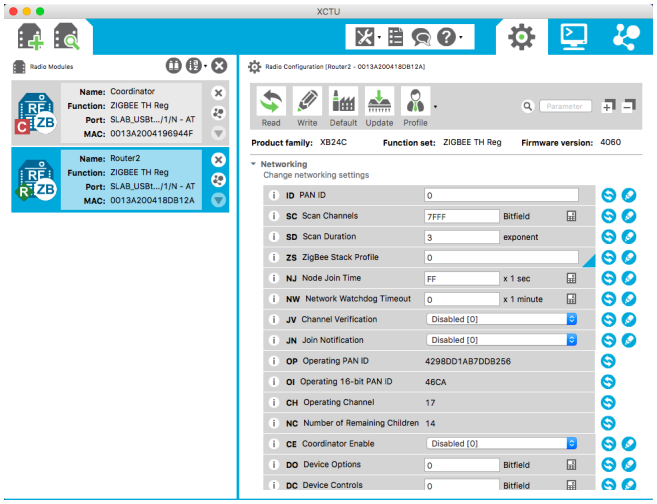
Hardware.

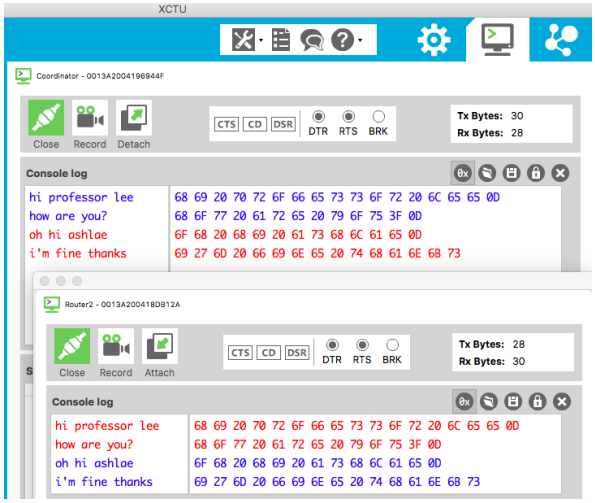
Since the goal of this project is to use motion sensor controls to explore dynamical systems, we will next address the hardware used. Our hardware device connects the LSM6DS3 inertial measurement unit to an XBee S2C network using the Arduino programming environment. The LSM6 sensors and XBee modules are selected for their flexibility in IoT applications such as this one. In this first prototyping phase of the initial system, a single LSM6 sensor is wired to an Arduino UNO, and using the LSM6 serial library [15], the module is programmed to continuously stream six pieces of data into the Arduino console: x-, y-, and z- coordinates for linear as well as angular acceleration.





Next, the XBee modules are configured to communicate via asynchronous serial UART using the settings: Product Family XB24C, Function Set ZIGBEE TH Reg, and Firmware Version 4060 [16]. For this step it is necessary to install FTDI drivers as well as the X-CTU and CoolTerm softwares. Using serial port configuration, two XBee modules are set up to operate in, respectively, Coordinator and Router modes. With the devices configured as such, communicaton is established between the two XBee devices (transmit message is blue, receive message is red).



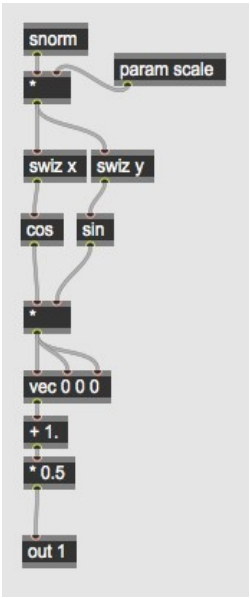


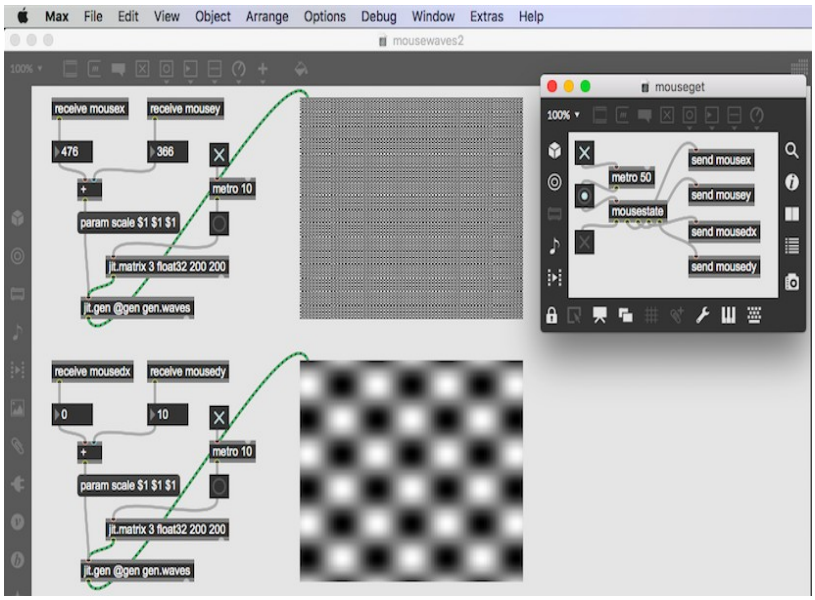
While serial communication is successfully established, we are not able to get our XBee components to work well with the Max/MSP environment. Hence, the interactive control system will make its introduction into our system in Part II of this project. In lieu of our wireless sensor controls, we have set up a placeholder in our software for future integration with the interactive environment. Hereafter, a computer mouse will serve to control the coordinate variations. Our further work continues with the exploration of the Max/MSP parameter space for a dynamical system which models harmonic oscillation.

Software.

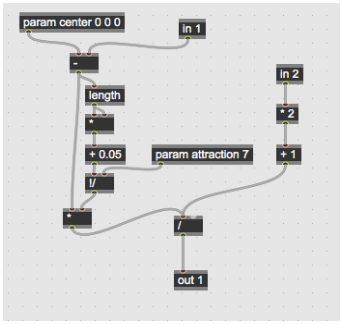
The software stage of this project models dynamical systems in the Max/MSP environment. For this, particle systems are chosen to represent the systems because of their versatility in applications of dynamic visualization [6]. The first step of our setup is to write a small program for receiving coordinate-based information into the Max environment, creating a placeholder for our sensor controls with a patch which polls for spatial *x*- and *y*-coordinates. In the demonstration below, the jit.gen code in the lower left image maps the mouse's spatial coordinates on the screen to the arguments of $\cos(x)\times\sin(y)$, normalized and scaled appropriately. Position $\langle x, y\rangle$ and velocity $\langle \frac{dx}{dt}, \frac{dy}{dt}\rangle$ are plotted in the lower right image as moving object processing matrices.

Sensor system test: Uses mouse coordinates as arguments for harmonic motion, then creates a vector field for the dynamics. Stores these values in a matrix and renders as a jit.gen object.

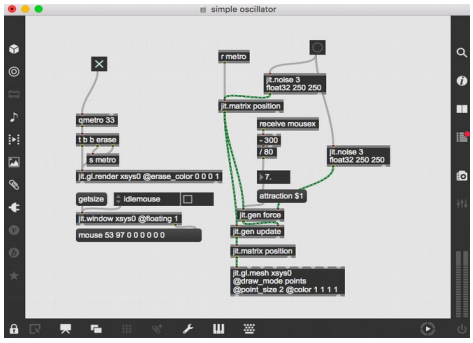




Next we build a system which incorporates dynamics using the inverse square law, a useful tool for describing statistical deviation from equilibrium in complex systems [17], [19], as well as a model of the power distribution for $\gamma = 2$. Using jit.matrix, we assemble a particle system and define a system of equations for the motion of the particles. We define a vector for the distance of any point in our system with respect to the origin and divide an effective attraction parameter by the square of that distance. Considering the equations of motion, we arrive at an equation of the form $\frac{d^2r}{dt^2} = \frac{k_a}{r^2}$, where k_a is an effective attraction parameter. This system is plotted graphically for an initial $k_a = 7$ using the jit.gen mapping function [20].



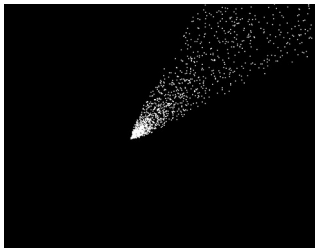
Dynamics for [jit.gen force]



System of equations for simple oscillator

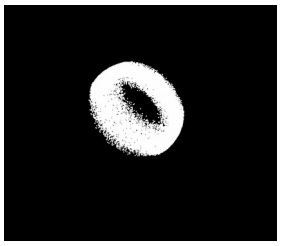
h) RESULTS

As we move our mouse over the screen, the resulting particle systems describe the phase space of the dynamical system as the attraction strength parameter k is varied. Six different regions of distinct behaviour can be seen over variations of k. For any $k \leqslant 0$, the particles diverge rapidly from the origin. The phase space stays in a ring- or seemingly torus-shaped limit cycle for $1<k<6$, and then moves into an intermediate transition region between $6<k<8$. Where $k>9$, an hourglass with a striated tail emerges. For $k>13$, there are almost no visible changes in behaviour, though the noisy particles around the manifolds increase to a greater density.

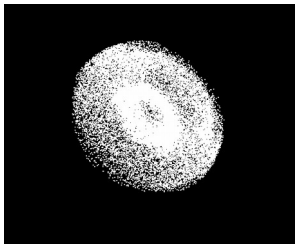


k=1

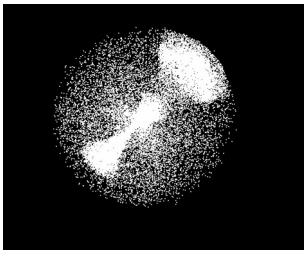
k=4



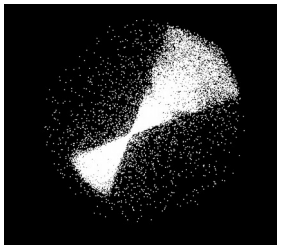
k=6



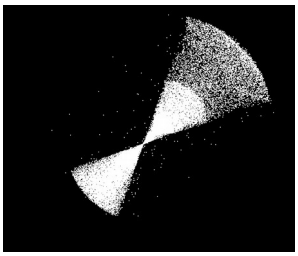
k=7



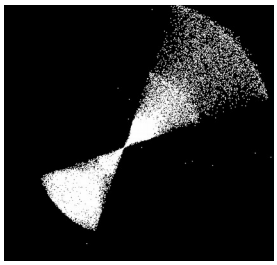
k=8



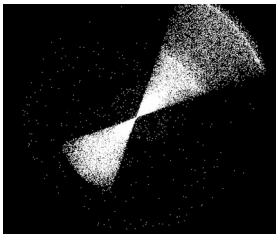
k=9



k=11



k=12



k=13

i) CONCLUSIONS & EXTENSIONS

For each of the regions seen in this phase space, several distinct behaviours are observed. As the particle systems are themselves moving, continuous variation of r can be treated as a gradual perturbation from the origin. These variations produced one type of unstable behaviour, and five types of stable behaviour. If we think of these phase spaces as cross-sections, or "slices" of a 3-dimensional object, we can imagine a continuous manifold where the behaviour varies over a surface which has holes, pockets, and varying regions of limit cycle density. Given that the images presented here are only 2-dimensional snapshots, their limitations are obvious in the use of studying multidimensional parameter spaces. In the time-evolution of the systems observed in this study, even viewing the motion of 2D particles on a screen brings the data to life, and features can be inferred which would otherwise not be possible in a static representation. Ideally, this project will be extended to include 3D and geometrical representations of the manifolds which describe these dynamical systems. For that, further work with and understanding of Max/MSP is required.

The next hardware portion of this project will be continued to incorporate the wireless controls of the XBees once their configuration and interfacing with the Max/MSP environment has been established. At that point, it can be expected that processing speed and graphics card capabilities will be the main limitations.

The next software portion of this project will be to include corrections to the forces acting on a driven oscillator, which will produce a more accurate picture of the network dynamics. Building the system in a this way will allow it to be easily scalable for a large number of nodes, and will increase the accuracy of the model, as well as its extensibility to a diverse number of scenarios. For example, this may be used as a research tool, an instructional tool, or in live performance scenarios, where VJs, DJs or dancers may want to input artistic content into an interactive system.

j) REFERENCES

[1] Y. Kuramoto, edited by H. Araki. *Lecture Notes in Physics, International Symposium on Mathematical Problems in Theoretical Physics.* Springer-Verlag, New York (1975).

[2] F. Rodrigues, T. Peron, P. Ji, J. Kurths. The Kuramoto Model in Complex Networks. *Physics Reports 610: 1-*98, (2016).

[3] J. Nitzbon, P. Schultz, J. Heitzig, J. Kurths, F. Hellman. Deciphering the imprint of topology on nonlinear dynamical network stability. *New Journal of Physics 19* (033029), (2017).

[4] D. Watts, S. Strogatz. Collective dynamics of small-world networks. *Nature 393:* 440–442, (1998).

[5] A. Barabasi, R. Albert. Emergence of scaling in random networks. *Science 286* (5439): 509-512, (1999).

[6] A. Witkin. An Introduction to Physically Based Modeling: Particle System Dynamics. (1997).

[7] J. Guerini, H. Peters. Julia sets of complex Henon maps. *International Journal of Mathematics 29*, (2018).

[8] H. Osinga, B. Krauskopf. Visualizing the structure of chaos in the Lorenz system. *Computers & Graphics 26*: 815-823, (2002).

[9] Strogatz, Steven H. Nonlinear Dynamics and Chaos. *Addison Wesley Publishing*, (1994).

[10] E. Lorenz. “Deterministic Nonperiodic Flow. *Journal of the Atmospheric Sciences 20:* 130-141, (1963).

[11] G. Medvedev. Small-world networks of Kuramoto oscillators. *Physica D 266*, 13-22, (2014).

[12] Y. Moreno, A. Pacheco. Synchronization of Kuramoto Oscillators in Scale-Free Networks. *Europhysics Letters 68:* 603, (2004).

[13] G. Filatrella, A. Nielsen, N. Pedersen. Analysis of a power grid using a Kuramoto-like model. *The European Physical Journal B* 61: 485–491, (2008).

[14] L. Zhu, D. Hill. Synchronization of Power Systems and Kuramoto Oscillators: A Regional Stability Framework. *ArXiv:1804.01644*, (2018).

[15] Pololu. LSM6 library for Arduino. *GitHub repository*. Posted 2016 January 19. Retrieved 10 May 2019. https://github.com/pololu/lsm6-arduino

[16] Jimblom, Toni_K. “Exploring XBees and XCTU”. *SparkFun.* Posted 12 March 2015. Retrieved 18 May 2019. https://learn.sparkfun.com/tutorials/exploring-xbees-and-xctu/all

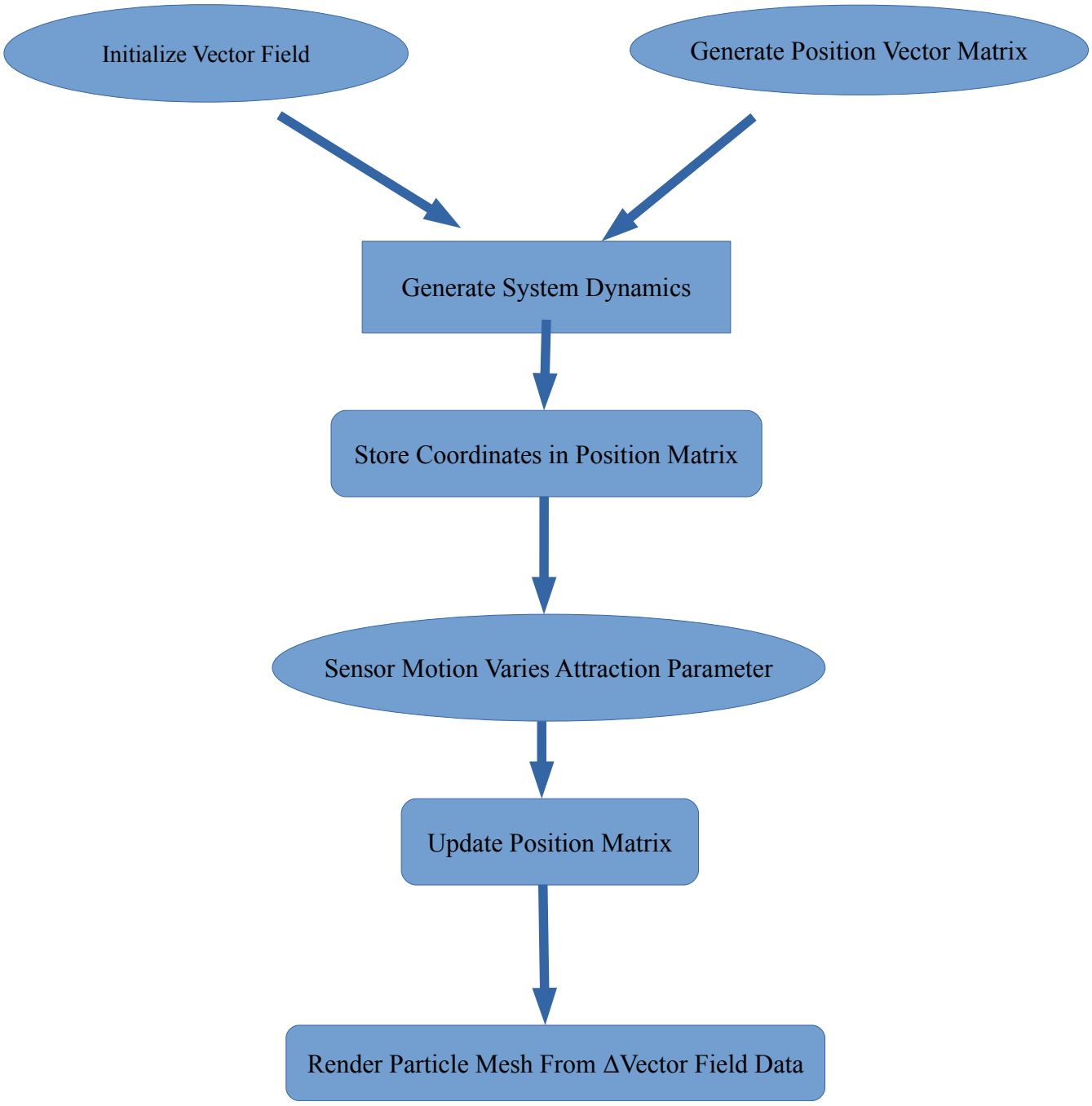
[17] A. Chakraborty, S. Easwaran, S. Sinha, "Deviations from universality in the fluctuation behavior of a heterogeneous complex system reveal intrinsic properties of components: The case of the international currency market", *Physica A 509:* 599-610, (2018).

[18] D. Alexander, F. Iavernaro, A. Rosa. Early Days in Complex Dynamics: A History of Complex Dynamics in One Variable during 1906-1942. *History of Mathematics 38, (*2011).

[19] R. Martinez-y-Romero, H. Nunez-Yepez, A. Salas-Brito. The two dimensional model of a particle in an inverse square potential: Classical and quantum aspects. *Journal of Mathematical Physics 54* (053509), (2013).

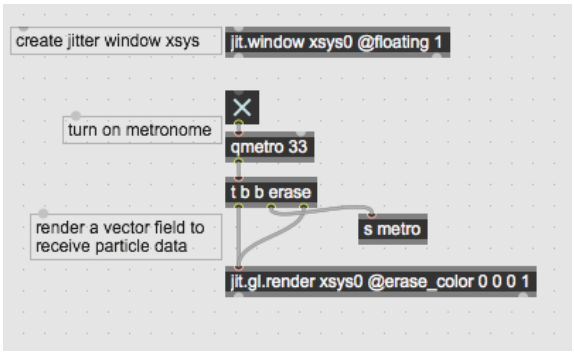
[20] F. Foderaro. “How to Build a Particle System in Max.” *YouTube.* Posted 4 January 2014. Retrieved 9 May 2019. https://www.youtube.com/watch?v=TRgX7rVgSAE

k) PROGRAM FLOW CHART

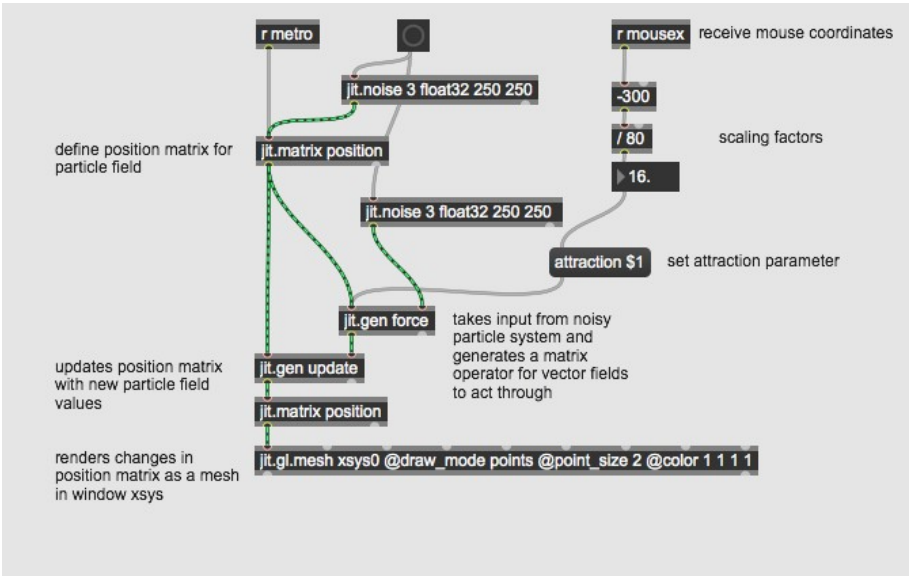


I) SOURCE CODE & COMMENTS:

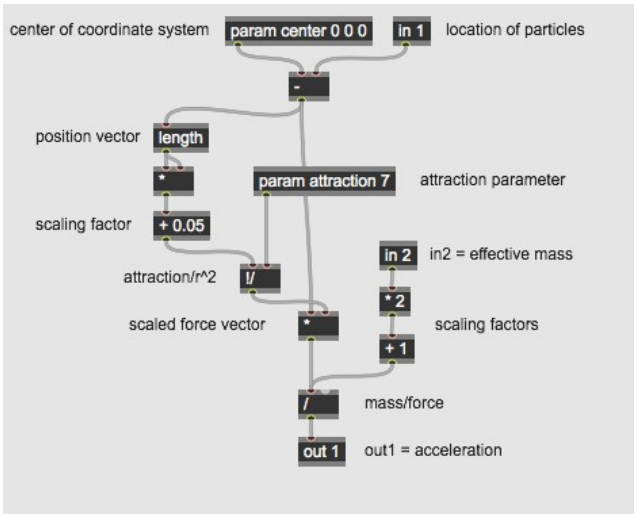
Max/MSP Setup:



Max/MSP Main Code:



Max/MSP System Dynamics:



LSM6 Arduino code [15]:

```
#include <Wire.h>
#include <LSM6.h>

LSM6 imu;                  //names the sensor

char report[80];

void setup()
{
  Serial.begin(9600);      //opens a serial connection at 9600 baud
  Wire.begin();

  if (!imu.init())         //if imu is off
  {
    Serial.println("Failed to detect and initialize IMU!");
    while (1);
  }
  imu.enableDefault();    //default state is to poll imu for data
}

void loop()
{
  imu.read();               //read coordinate data

  snprintf(report, sizeof(report), "A: %6d %6d %6d    G: %6d %6d %6d",
    imu.a.x, imu.a.y, imu.a.z,      // get x, y, x linear acceleration values
    imu.g.x, imu.g.y, imu.g.z);     // get x, y, z angular acceleration values
  Serial.println(report);

  delay(100);               //wait 100ms before updating
}
```