Senior Design Final Report

Small Unit Drone Optimization (SUDO)

Version 1.0 - 04/13/2023

Team Members:

Jonathan Aguirre

Lloyd Castro

Jean Paul Espinosa

Peter Han

George Hernandez

Hugo Izquierdo

Raymond Martinez

Bruck Negash

Jesus Perez Arias

Faculty Advisor:

David Krum

Liaisons:

Dr. Peter Khooshabeh

Dr. Eric Holder

Dr. Paul Fedele

Table of Contents

1. Introduction:

1.1. Background:

The Army Research Lab (ARL) is the US Army's sole foundational research facility that is focused on scientific discovery and technological innovation for the advancement of the US Army's chances of surviving and winning future conflicts. The ARL has regional sites spanning West, South, Central, and Northeast of the United States . We worked with ARL West, who specialize in human interference and human information interaction. ARL West presented us with an idea on how to digitize human perception into a visual model that can be used to predict and create data. We were also provided with data in the form of human perception models aggregated from decades of research, with the end goal of interpreting and using these models to create a program which can predict the detection distance from which an SUAS (Small Unmanned Aerial System) could be detected by enemy troops. We then used a 3D environment to visualize this detection distance and add additional parameters to our model.

What is the human perception model? Briefly, the human perception model is an overview of how the human senses work together to perceive things in real space. The human senses that were highlighted in this model were specifically sight and sound. The dataset we were presented with showed how environments contribute to sight and hearing of certain objects, and we utilize that data to determine what it takes to see and hear a drone (SUAS). Using this premise, we will be creating an updated integration of said module into a python program to create a visual representation of the model in the Unity Engine. With this Unity visualization, we aim to create a geographical simulation of a mission-area and show how a drone can be sent into that mission area with the smallest detection possibility.

This program will be run in Python in unison with Unity, in order to create visual data representations of how drones will be detected within a given situation.

1.2. Design Principles:

The program will be created in Python, using VBA (Visual Basic Analysis) code that we have extracted from the Auditory Model, and translated to the Python language with modern programming principles. The purpose of the Python program is to take in drone distance inputs, and output data that will aid in determining if the drone will be detectable or not with the given inputs. Once the Python program is initialized, it will then work with the Unity engine to render the 3D environment using various images from landscapes. Unity will also create an interactive simulation where we can test detectability with drones in an enemy location. Using a modern programming language like Python, as well as a modern visualization of the original data set allows us to move away from the ARL's Auditory Detection excel sheet in a way that makes it easier for troops on the ground to interact with, and thus more accessible to the end-users.

1.3. Design Benefits:

In using the Python programming language, we have created a base for the updated detection macros to be easily translated into a usable format, and allows us to access them through our development environments. Python is one of the most popular modern programming languages, and will probably remain that way for the foreseeable future. Thus, our success in translating these VBA macros into an approachable and easily understandable language is a major improvement in versatility and modularization over using a hard-to-understand programming language like VBA.

In Using unity we are able to replace the visual aspect of the human perception model into a 3D interactive one. In conjunction with the Python program, Unity will be able to take the data that the program outputs and create a visual representation of the probability that a drone will be detectable, demonstrating affecting factors such as sightlines. This approach to visualization will give the end-user (troops on the ground) a better insight into how the input parameters affect the detectability of the drone, as well as a better picture of the overall situation on the ground. Unity also allows the end-user to take scans of locations and create a 3D environment based on the actual mission-area in order to best simulate operations in hostile terrain.

1.4. Achievements:

Over the course of the academic year, our team has been deciphering and interpreting the flood of data and documents that the ARL team has provided us. We worked together to understand the background knowledge and math that goes behind the human perception model that we were tasked with improving. We have learned how to dissect and translate VBA code, understand how the language works, and also how to properly translate it into another language, while retaining core functionality.

After spending so much time understanding, interpreting and translating the functionality of the human perception model, we were able to create a program that is much cleaner, easier to use, and expandable into other forms of usage instead of relying on independence from a single user-interface to interact with the program. Further, our Unity program takes our achievement further by allowing for a visual real-time detection module which can allow end-users without much technical experience to ascertain the likelihood of a drone's detection when conducting reconnaissance in enemy zones.

2. Related Technologies:

2.1. Existing Solutions:

We have scoured the web and looked through various publications, but have yet to find any software that comes close to our end-goal with this project. While there are some software solutions that exist for allowing drones or other pieces of hardware to detect humans in real-time (such as yoloV5), all of them simply take the approach of measuring the machine's efficacy in terms of its own capability. We have actually gone from the opposite side, and incorporated the capabilities of human perception in order to make a much more accurate model that , while able to work as a model independently from on-board sensors, would be greatly improved by their use.

The closest thing akin to our program would be Unity scripts pertaining to line of weight and detection. These scripts would have similar goals to ours, but are nowhere near as nuanced and mathematically-sound as the decades of research which we were provided, which among many other benefits, allows for the dynamic changing of parameters and inputs.

2.2. Reused Products:

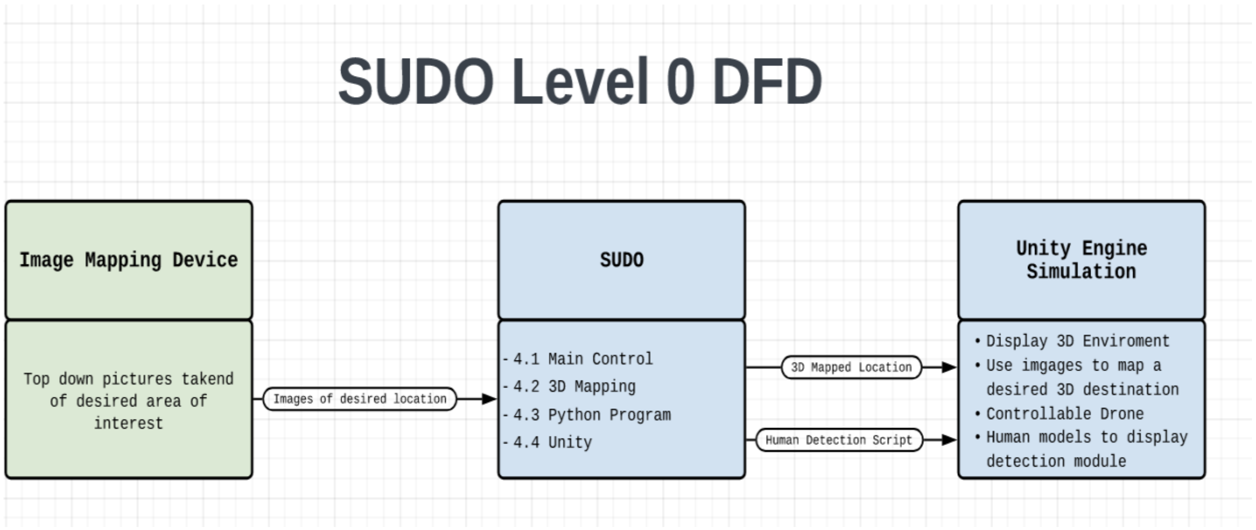
The program used to run calculations of the model is using Python with a few libraries installed to make the calculations which are sent through Unity to create the visual simulation.

3. System architecture:

3.1. Overview:

The architecture for SUDO can be broken down into three main factors: the Image mapping device, SUDO itself and the Unity Engine Simulation.

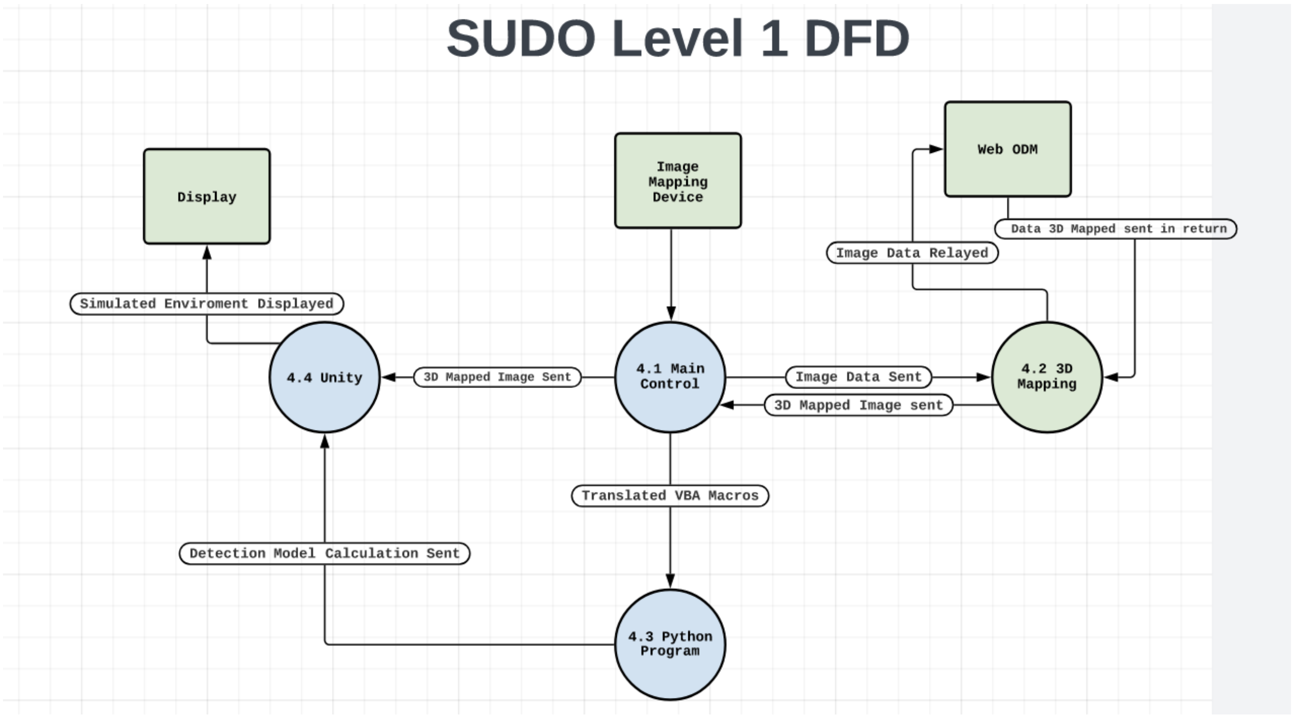
Here is a diagram (DFD level 0) that shows how this architecture works at a high level:



- **Image Mapping Device (User):** This is the part of the architecture that initializes the creation of the 3D simulation. The mapping device is employed by the user to take images of a location of interest. Those images are then sent through SUDO to be processed and utilized for the simulation in Unity.
- **The Framework (SUDO):** The main goal and frame of the project, providing the backbone to create the Unity engine simulation. The framework takes in the user's images, then processes those images for use in the Unity engine. It also maintains the Python program which runs the calculations of the human perception model so that the unity simulation can display those calculations and visuals to the user.
- **Unity Simulation:** The unity simulation is the conclusion to our goal. It takes in from the SUDO framework and displays the simulation to the user. The Unity simulation takes in the 3D renditions of the provided images to create the virtual environment that it can use. The Python program then outputs a script that the unity engine can use to display the visual version of the human perception model, which lets the user know the maximum distance a drone can be employed before detection.

3.2. Data Flow:

Here is an overview of the Framework as a system, and how it connects to Unity, incidentally, this is also our DFD level 1:



There are five major modules in this system. They are described in more detail in section 6. Here is a brief overview of them:

3.4.1. Main Control: The Main Control module will be responsible for securely combining all the data that is necessary for each file transfer and visual display. It securely reveals the sum of every module for the user to see.

3.4.2. 3D Mapping: The 3D Mapping module will be responsible for creating a 3D environment based on the images the drone collects as well as coordinates from the general area covered using ODM/WebODM.

3.4.3. Python Program: The Python program module is responsible for processing auditory input parameters and generating accurate predictions of a person's ability to detect a target Small Unmanned Aircraft System (SUAS) based on those parameters. The program outputs a range of detectability "zones" based on possible hearing-quality thresholds of the observed person. These ranges are visually represented within the Unity Engine to show the maximum distance that an SUAS can approach a target before being detected.

3.4.4. Unity: The Unity module then takes all the data that is processed such as the 3D mapped location to render the simulation world. Then, it takes in the script of the Python program and calculates the variables necessary to create a close estimate of the detection distance needed in order for the drone to be spotted.

3.3. Implementation:

The project was split into four sections to allow for efficient development: VBA macro translation, Python Program, user interface and experience, and the Unity simulation. Each section plays a key role in presenting the progression of the project.

3.3.1. VBA Macro Translation

Initially in the development cycle of SUDO, we were presented with a lot of research and scientific models in regards to human perception. Our goal was to take this accumulated research and logic and turn it into a modern program. From the beginning, we reviewed line-by-line through this old VBA program and decipher math and physics formulas involved, and how to best make use of the inputs being fed to the program to output similar or better from the new software.

3.3.2. Python Program

With the macros translated into a pseudo-code, we proceeded to create a program in the Python language which could efficiently process the information and calculations in the same way the VBA macros were. This task included how to best make use of the excel data, while simultaneously moving away from being tied to strictly using this excel sheet in the future. We split the program into three separate files, creating a Python dictionary for necessary data, and cleaned all redundant old visual basic code. This resulted in a lean, efficient program performing the necessary calculations to export to the Unity engine while having scalability in mind, therefore displaying them in a visual way.

3.3.3. User Interface/User Experience

The user interface is designed in Unity, since the user will open the executable program to see a menu which will allow for the changing of some of the program parameters when running the simulation. Users will also be able to create a 3D model environment if needed and will be able to run the script that applies their choses values via simple button clicks.

3.3.4. Unity Simulation

The simulation is based in Unity, and uses all the models to incorporate an interactive simulation representing how far a drone needs to be from a human target in order to stay out of the person's range of detectability. It uses the 3D rendered environment to move around in a virtual sandbox, and the Python program to derive the calculations needed to visualize the drone's detectability likelihood. This is done by letting the user easily navigate a simplified UI in order to apply any necessary values and will let them choose any 3D object they have previously collected elsewhere.

4. Conclusions:

4.1. Results:

Our team created a modern version of a human perception model decades in the making by the Army Research Lab. While it is not currently dynamic in the sense of the drone making course-corrections based on sensor-data, it is the stepping stones towards a more effective and interactive model.

We combined the operations of the Python program with the Unity environment so they can transfer and share data with one another. In this way we can best represent how human perception values affect the drone's detectability range and likelihood in the simulation.

In the python program, our team was able to calculate the detection distance with various input values. However, we had to modify the detection distance by scaling it due to issues with some of the variables in the program. In doing so, we can obtain the result we want to test with our Unity simulation until the functions in our python program are fixed. One possible issue is with referencing and memory allocation of some of the lists that are passed through various functions in a single process. Another issue may lie in high level physics formulas that were mistranslated due to conversion from global to local scope.

In the Unity program, our team was able to create a simple UI that will let users add or change any parameters needed in order to get a detection distance output. The vast number of input fields and checkboxes let the user adjust the respective parameters accordingly. After the user applies the values, they will be saved into a JSON file and be placed as necessary into the Python script. Users will also have the option to create a 3D environment based on their images chosen via connection to the ODM site by Docker, or they can import an already made 3D environment based on an object file previously made. The simulation will then use the detection distance output in order to adjust the detection ring visually and accordingly depending on a few factors.

4.2. Future:

We planned to incorporate the yoloV5 software in order to detect objects in real time, but we did not get to this point. While we translated the detection model from the VBS to Python, we realized that the limitations of our program was that we could not account for the actual technology and camera quality onboard a drone within the timeframe we had. With a better camera usually comes increased size/weight, which would increase the sound profile of the drone, and thus increase the distance in terms of proximity to a human target before being detected (meaning it could not get as close as a smaller, quieter drone before detection). So we decided to work on the detection model itself, and leave the hardware testing to next year's group. The Unity portion can also be expanded on as the input fields and menus are set up in a dynamic format that will be of use when translating other VBS models to Python. Future groups will also be able to expand upon the detection model by making the visuals more straightforward and informative during runtime.

5. References:

Intaratep, Nanyaporn, et al. "Experimental Study of Quadcopter Acoustics and Performance at Static Thrust Conditions." *Aeroacoustics Conferences*, American Institute of Aeronautics and Astronautics, 2016.

Barten, Peter G.J. "Formula for the Contrast Sensitivity of the Human Eye." 2004.

US Army Human Engineering Lab, Garinther, Georges R, et al. *Design Standards for Noise: A Review of the Background and Bases of MIL-STD-1474(MI)*, US Army, 1975.

Airborne Drones. "Drone Noise Levels." *Airborne Drones*, 24 Oct. 2021, www.airbornedrones.co/drone-noise-levels/.

Lee, Yaesop, et al. "Archangel Dataset: UAV-Based Imagery with Position and Pose Metadata." *Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications IV*, 6 June 2022, https://doi.org/10.1117/12.2618636.

Jungkunz, Patrick. "Modeling Human Visual Perception for Target Detection in Military Simulations." *Naval Postgraduate School*, Dudley Knox Library, 2009.