Space Object Tracking from CubeSats utilizing Low-Cost Software Defined Radios

AE 8900 MS Special Problems Report
Space Systems Design Laboratory (SSDL)
Guggenheim School of Aerospace Engineering
Georgia Institute of Technology
Atlanta, GA

Author:
Alex Mealey

Advisor:
Prof. Brian C. Gunter

December 15, 2023

Space Object Tracking from CubeSats utilizing Low-Cost Software Defined Radios

Alex S. Mealey*
Georgia Institute of Technology, Atlanta, Georgia, 30332

Dr. Brian C. Gunter†
Georgia Institute of Technology, Atlanta, Georgia, 30332

This paper focuses on the implementation of a low cost software defined radio in small satellite applications for resident space object tracking. This research has the objectives of gaining flight heritage for low cost SDRs, utilizing an SDR as a navigation device and to employ an in orbit SDR as a satellite tracking device by performing orbit determination of transmitting satellites. The integration of a low cost SDR, known as a HackRF, onto the OrCa2 satellite will be discussed in detail in this paper. This integration will allow for rigorous testing of the SDR and will increase the knowledge of how the HackRF will perform in orbit. An SDR setup, centered around a HackRF, was tested as a navigation unit, through both a GNSS simulator and real time signals, which verified that it can indeed provide a timely and accurate navigation solution. The feasibility of using an on board SDR to determine the orbits of transmitting satellites was tested using Doppler-only initial orbit determination through Matlab simulations and real time data. The Matlab simulations provide a framework to test and develop Doppler-only initial orbit determination further and show the possibility of orbit determination from in space. However, complications with the recording method prevented the successful testing orbit determination of a transmitting satellite with real time data.

*Graduate Research Assistant, Space Systems Design Laboratory
†Associate Professor, Space Systems Design Laboratory

I. Nomenclature

ϕ = phase
Δx = distance between antenna
θ = angle of signal arrival
ρ = range
r = position vector
v = velocity vector
$\dot{\rho}$ = range rate
f_i = measured frequency on arrival
f_0 = transmit frequency
μ = gravitational parameter
Subscripts
GS = Ground Station
S = Satellite

II. Introduction

In most spacecraft applications, knowledge of the spacecraft's position and velocity is key information. This knowledge is often crucial in science and communications for spacecraft operations. Thus, most spacecraft utilize a Global Navigation Satellite System (GNSS) receiver to determine their position and velocity in space.

However, typical market GNSS receivers can be expensive and difficult to integrate, making them a challenge to implement onto spacecraft. These challenges are heightened further when applied to a CubeSat, a smaller, more modular spacecraft in which cost and ease of integration are even more important metrics. This led the authors to ponder how recent developments in radio technology could be utilized to mitigate the current challenges of GNSS receivers.

Software Defined Radio (SDR) technology has existed since the 1970s but has become more accessible by the public recently with off the shelf SDRs becoming increasingly more cost effective. An SDR is a radio in which some or all of the physical layer functions are software defined. Thus making an SDR easier to modify and much more suited for cross-functionality. This technology grants SDRs with the ability to use wide ranges of frequencies, sampling rates, bandwidths and more. The cross-functionality of SDRs opens exciting opportunities for a single SDR to serve multiple purposes on the same mission. This could include receiving commands, down linking data, GNSS signal reception and orbit determination of other satellites. This paper will explore the utilization of SDRs in CubeSat applications by integrating an off the shelf SDR known as a HackRF onto a Georgia Tech CubeSat mission.

The specific application discussed in this paper is the Orbital Calibration (OrCa2) mission. OrCa2 has a primary objective of increasing fidelity of orbital and optical models by having optically calibrated panels that can be viewed while in orbit with ground telescopes. In the scope of this mission, the SDR will be an experimental payload with two objectives. 1) Provide navigation data (position and velocity) using the Global Positioning System (GPS) and 2) Perform orbit determination on other satellites using Doppler frequencies and angle of arrival (AOA). From this point forward, the paper will be split into two sections each addressing the two objectives previously stated: Navigation unit and Orbit Determination (OD).

III. Motivation: Navigation Unit

For almost any satellite mission, knowledge of the position and velocity of the vehicle is crucial information thus highlighting the importance of a reliable navigation unit. Navigation units are also crucial for CubeSats, however CubeSats typically emphasize efficient design both in terms of time and finance. The typical GNSS receiver from a commercial vendor can cost upwards of $8,000.

The SDR receiver has the potential to ease both of these issues, with a price point approximately an order of magnitude less than commercial receivers and parts that can be ordered and shipped in less than a week. Plus, due to their flexibility, a singular SDR has the potential to replace both communications and navigation components further increasing the efficiency of the CubeSat.

Additionally, the lack of flight heritage for off the shelf SDRs creates open questions in terms of how they will perform in orbit and hold up against the harsh space environment. This includes being able to withstand the large temperature variations and increased radiation exposure. Through OrCa2, there is hope to gain fight heritage and answer these questions.

IV. Methodology: Navigation Unit

Implementing SDR technology onto a satellite was a challenging effort for multiple reasons. These reasons include the lack of flight heritage for off the shelf SDRs and the wide array of options in terms of both hardware and software configuration. In this section, the configuration options for the SDR system will be described and explored, along with the integration of the SDR system into the rest of the spacecraft and the testing of said system.

A. Hardware Components

When constructing an SDR system there are four components to consider: The SDR peripheral, the SDR computer, the oscillator and the antenna. A diagram of these four components and how they interact can be seen in figure 1. What follows is a discussion on the four components in terms of their functionality, important metrics, and how each component was chosen for the OrCa mission.

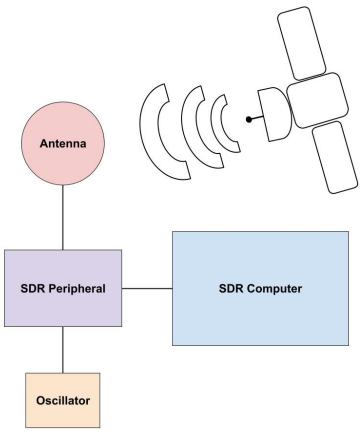

Fig. 1 SDR Components

also be considered for flight applications. Table 1 breaks down the important attributes of both devices.

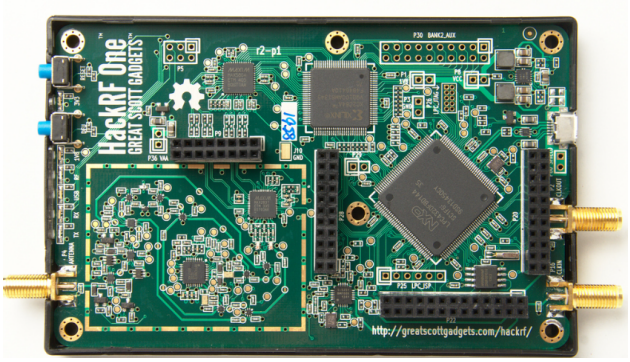

Fig. 2 HackRF One

1. SDR Peripheral

Firstly there is the SDR peripheral, which is often referred too as just the SDR. This is technically incorrect, since a complete SDR system requires a computer to configure and command the SDR peripheral. The peripheral includes a transceiver, an analog front end, a mixer, a synthesizer and several other traditional radio components that are configured by the SDR computer.

Due to SDRs becoming popular among amateur radio groups and their utility in testing/prototyping radio systems, there is a large demand for cheap, off the shelf SDR peripherals. This demand has resulted in a wide array of SDR peripherals being offered to the public with different price points and specialties. Some of the most popular SDR peripherals today are the HackRF one, the Ubertooth, the YARD stick and the RTL-SDR dongle. These SDRs are readily available and are relatively cheap with a price range of roughly $30 to $500.

When considering which SDR to employ for use on the OrCa mission, the most important metrics were: 1) Low price point to enable the purchase of several for testing 2) Small form factor so it can fit well onto the CubeSat and 3) Wide frequency array for cross-functionality

The two SDR peripherals that were used for this application were the RTL-SDR dongle and the HackRF one. The RTL-SDR dongle was chosen for its low price point (about $35), small form factor (about the size of a USB memory stick) and excellent clock stability which enables GPS lock (will be discussed later in the Oscillator subsection). The HackRF one was chosen primarily for its cross functionality with the ability to both transmit and receive a wide frequency range of 1 MHz to 6 GHz. Ultimately, the HackRF one was chosen as the flight unit in hopes to gain flight heritage on such a multi-use device. An image of the HackRF is provided in Figure 2. However, the RTL-SDR dongle was still used as a helpful testing device and could

2. SDR Computer

The SDR computer, which will be henceforth referred to as just the computer, is the second part of the SDR. It works in junction with the peripheral to receive and transmit radio signals. The computer is used to configure the settings of the peripheral such as sampling rate, bandwidth, and frequency as to meet the requirements of a specific application. The computer also processes both the signal and the data from or for the transmission.

The software required to work with an SDR peripheral is relatively small and doesn't need to be memory intensive. However, the software specifically for GNSS signal reception and position lock is more intensive. Thus, there are basic requirements for the SDR computer. The computer must have at least 4.5 GB of storage (preferably more to store data from tests) and 4 GB of RAM in order to successfully lock onto GNSS satellites.

There were several computers considered and used, some for only testing and some for flight units. For testing purposes, a Mac Book pro was often employed. The computer is well beyond the aforementioned requirements and is an excellent testing device to ensure component capability or to gather data. However, the laptop obviously does not have the form factor to integrate into a CubeSat.

The BeagleBone Black is an embedded Linux device that is being used as the primary on-board computer (OBC) for OrCa2. Thus, for simplicity, a BeagleBone was the first computer attempted for SDR applications. This quickly led to storage issues but these could potentially be remedied by mounting external storage devices. The next issue was the lack of RAM on the BeagleBone black. It was discovered that the required software for GPS position fix simple could not run on a BeagleBone without major modification.

Thus, the secondary OBC on the mission was considered: The Raspberry Pi. Raspberry Pies are a very popular single board computer used for wide arrays of applications

Table 1 HackRF vs RTL-SDR Attributes

	Size (mm)	Mass (grams)	Clock Stability	Transmit Capability?	Price (USD)
HackRF	$124x80x18$	100	$20ppm$	**Duplex**	350
RTL-SDR	$69x27x13$	184.4	$1ppm$	**Simplex**	34

from automated bitcoin miners to embedded network devices. The OrCa team initially added the Raspberry Pi to the mission to control an on board imager meant for optical attitude determination. However, the Raspberry Pi meets the criteria for the SDR payload as well making it a natural transition.

There are several versions of the Raspberry Pi, three of which were used extensively in the research for OrCa. The first is the Raspberry Pi 4 (RPI4). The RPI4 model B was at the time the newest Raspberry Pi and boasted a RAM of up to 8 GB. Additionally, the RPI4 utilized an external micro SD card for storage meaning the RPI4 can have as much storage as the micro SD card (up to 1 terabyte and rapidly increasing). The RPI4 comes with several useful interfacing and management capabilities including HDMI ports, 4 USB ports, an Ethernet port, USB-c power port and a cooling fan. While these features make the RPI4 an excellent testing device, they also increase the size and make it a poor option for a flight unit. Luckily, there are several Raspberry Pi computers that have a much smaller form factor.

The Raspberry Pi Zero W was considered due to it's extremely small form factor and price point, however the limiting 512 MB of RAM prevented successful use with SDR. The Compute Module 4 (CM4) however, has the same processor as the RPI4, the ARM Cortex-A72 processor, thus allowing for use with GNSS and SDR applications. The CM4, shown in figure 3, has a much more compact form factor than the RPI4, as seen in the figure below, making it great for spaceflight. However, the limiting form factor does make interfacing with the board more difficult so an I/O board was used for early testing and integration.

3. Oscillator

An oscillator is required when demodulating GNSS signals. The oscillator generates a stable reference frequency which allows for GPS signal correlation. Specifically for spaceflight, it is recommended to use a temperature controlled crystal oscillator (TCXO) since it will not drift with dramatic changes in temperature as are common in orbit. The main requirement of the TCXO is a frequency stability of 1 ppm, since this is required for GNSS signal processing.

Most SDR peripherals come with an in board oscillator, however most of them are not stable enough for GNSS signal processing. Luckily, the HackRF has a CLKIN port which allows for an external oscillator since the internal

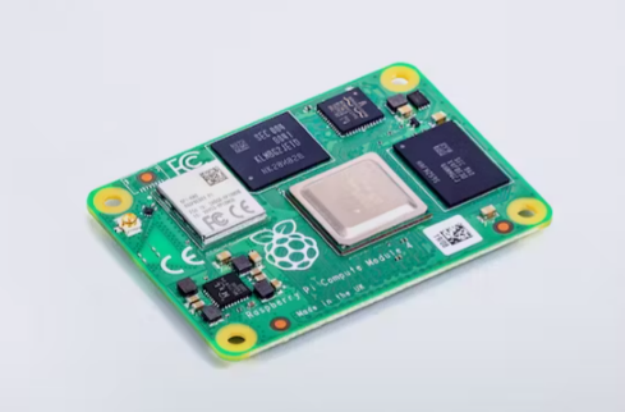

Fig. 3 Raspberry Pi Compute Module 4

one is not accurate enough. The RTL-SDR dongle has an on board oscillator with a stability of 1 ppm, making it good enough for GNSS signal processing.

The only oscillator that was successfully used in conjunction with the HackRF is the LOTUS temperature controlled crystal oscillator (Figure 4). Using this TCXO is very simple and can be viewed as plug and play. The output signal is transmitted via a SMA cable that runs to the CLKIN port of the HackRF. The TCXO is powered via a +3.3v and ground terminal. The Lotus TCXO is quite simple and had a relatively low form factor as can be seen in the below figure. This TCXO provides a stability of 0.28 ppm, significantly better than the minimum requirement for GNSS signal processing. This TCXO costs $119.00

Fig. 4 Lotus Communications Systems TCXO

4. Antenna

The final component of the SDR system is the antenna. Antennas are used to both receive and send radio waves by converting electromagnetic waves to electrical current or vice versa. GNSS signals are typically pretty weak by the time they make it to a receiver thus having an antenna with a significant gain was important. It should be noted that both the HackRF and the RTL-SDR dongle can send bias voltage to the antenna to increase gain. Specifically for GPS signal processing, a GPS L1 antenna is required.

There were two antennas used extensively for this research. Firstly, a magnetic GPS L1 antenna was used primarily for testing. This antenna is cheap ($10) and very easy to use for testing and data gathering. However, this antenna can not be considered for the CubeSat since we have very sensitive magnetic instruments on board for attitude determination and control systems (ADCS) and the magnetic field from the antenna might complicate these operations.

Instead, on flight we will use the TW1829 Dual-Band Antenna by Tallysman (figure 5). This patch antenna is specifically designed for GNSS and has a maximum gain of 26dB. Additionally the Tallysman is a small form factor and has flight heritage from previous missions from the same lab as OrCa.

Fig. 5 Tallysman Ceramic Patch Antenna

B. Hardware Assembly

Assembling the components for the GNSS receiver is not very complicated but does require some hardware modifications.

The required connections for the assembly to work include wiring for power supply, data streams and RF transfer. The diagram in figure 6 shows the necessary cabling for the assembly. This diagram also includes the cabling for other payload components, but for the sake of this paper CAN hat and imager can be ignored.

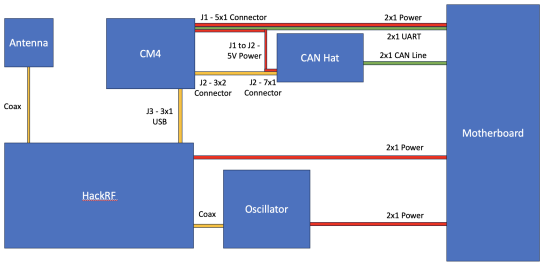

Fig. 6 Cabling Diagram for SDR System

The HackRF is powered by a USB cable that also transfers data between the SDR computer and peripheral. However, the research team was worried that a micro USB port would not maintain a secure connection during the launch of the satellite. Thus, the plan is to make a more permanent connection by removing the micro USB port and soldering the wires directly. This also allows one to breakout the USB wires so the data streaming can connect to the CM4 and the power can be supplied directly from the motherboard.

In a similar manner, in order to make the connections to the TCXO more robust, wires were soldered directly to the voltage and ground pins as shown in figure 7. In the future, it would be advisable to do so with heat shrink wrap as well for an extra secure connection.

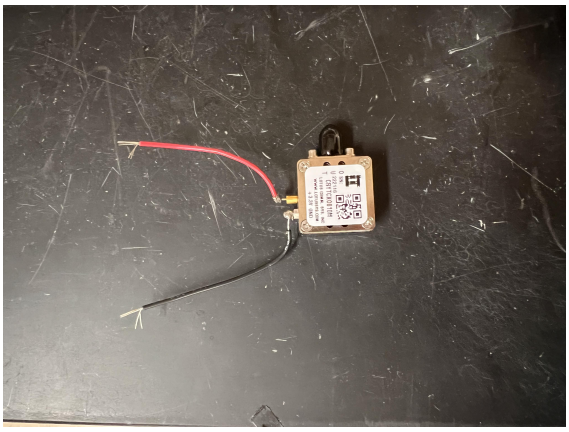

Fig. 7 Soldered Connection to TCXO

The HackRF comes in a plastic case that protects the interior components. It is advisable to keep this case on if outdoor testing is expected. However, for the flat sat setup and the flight assembly, shedding the plastic casing makes the HackRF better suited for the harsh space environment. Shedding the plastic casing simply requires sliding a thin rigid object in between the top and bottoms layers and increasing the separation. On this project, a guitar pick was used.

C. Software Configuration

The SDR system requires software to receive commands, manage the SDR, process signals, and export data products. This software will run on the payload OBC, in this case the Raspberry Pi CM4. Each of these tasks will be elaborated on in the following sections. The software developed for this project is stored in one of two places. All software involved in managing the CM4, task management or communications can be found in the GitHub repository PayloadProcessor in the secure GitHub. The software associated with GNSS-SDR can be found in the GitHub repository gnss-sdr-rpi.

1. SDR Management

The managing of the SDR includes sending power to the SDR peripheral, running system checkouts and shutting the system off gracefully. This is done via Python scripts that run on the CM4. When the CM4 receives power it will run all functions within its own rc.local file thus a script was added to start up the SDR peripheral. This includes a system checkout looking at factors such as available storage and internal temperature of the CM4. The checkout also includes checking for hardware connections and will ensure the SDR peripheral and TCXO are connected before running any experiments. Once the system has passed its checkout, the CM4 will then communicate with the main OBC to determine what mode it is in. There are two modes the CM4 can be set into while in orbit, these are science and downlink. Once the mode is confirmed with the main OBC, if in science mode the OBC will then begin running the SDR system to try to gain a position and velocity fix. Shutting the system off gracefully involves receiving a shut off message from the main OBC, sending shut off command to the payload processes (include the GNSS receiver) and then shutting off the entire CM4.

2. GNSS-SDR

The signal processing is done by an externally created software package called GNSS-SDR, which stands for Global Navigation Satellite System - Software Defined Receiver. GNSS-SDR is an open source free software package, with an active community of users, available in most common operating systems. This purpose of GNSS-SDR, as its name implies, is to utilize SDR components to create a GNSS receiver. The data flow for GNSS-SDR is illustrated in Figure 8. There is not available technology to digitally process signals at the frequency sent by GPS satellites, meaning an RF front end is still required to down sample the signal, filter and mix signals, and then deliver quantized raw signals to the computer. At this point, GNSS-SDR begins the process of obtaining receiver data and potentially a position/velocity fix. The software will commence digital signal processing, signal acquisition and

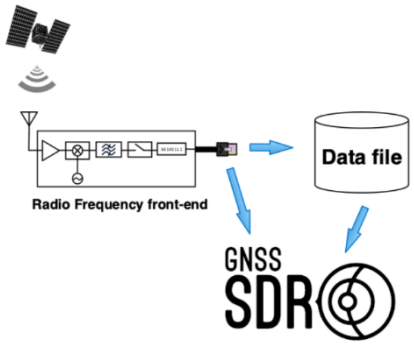

Fig. 8 Data Flow from Satellite to Computer

tracking of present satellite signals, decoding the navigation message and computing the navigation solution.

The way GNSS-SDR accomplishes this is by creating a signal flow graph, similar to GNU radio. This flow graph determines how incoming signals are conditioned and utilized in the navigation solution. The generic framework for this flow graph is shown in Figure 9 . All of these signal processing blocks can be altered easily by the user through a single configuration file. Modifying this configuration

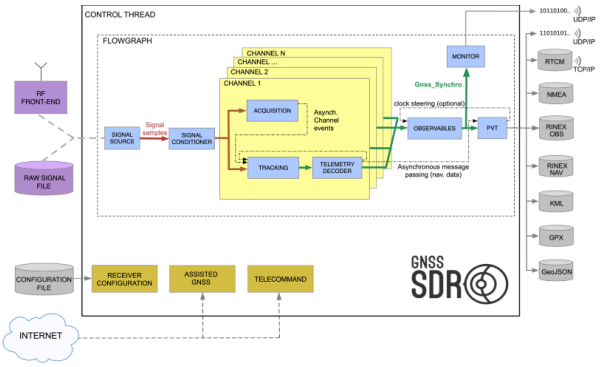

Fig. 9 Framework for GNSS-SDR Signal Flow

file is as simple as editing a text file. Some of the common configuration options considered for this research, and how they were modified, are included below.

1) Antenna Bias: How much voltage is being sent from the SDR peripheral to the antenna? The patch antenna discussed earlier can handle a voltage from 2.5 to 16 volts. The HackRF can only supply an antenna bias of 3.3 volts thus the bias was set to true to supply 3.3 volts to the antenna.
2) Doppler Shift Maximum: What is the maximum change in frequency due to relative velocity the receiver could see? The incoming GPS signals will almost always have a frequency different from the original transmit frequency due to the Doppler shift (this phenomenon is discussed further in later

sections of this paper). Due to this fact, when looking for signals, GNSS-SDR sets a range of expected maximum and minimum frequencies and searches that window. The size of this frequency window had to be altered for the in orbit case due to a change in maximum and minimum relative velocities.

3) Dynamic Model: How is the receiver moving? This metric can be set to account for either a moving or a stationary receiver.
4) Number of Channels and Channels in Acquisition: These configurations determine the maximum number of satellites targeted in a navigation solution and the maximum number of signals that will be acquired at once. These metrics are important because the more satellites in a solution the more accurate the data will be, and the more signals being acquired at once will increase the speed of the process. However, most computing platforms, especially embedded devices, will have limits on these metrics due to limited processing power. When the RF front end is configured to take in more data than the attached computer can handle, GNSS-SDR will print '000000' repeatedly to alert users. For the CM4, it was found that setting the number of channels to 8 and the number of channels in acquisition to 4 produced the best results.

These are just some of the many configuration options available through GNSS-SDR. Another configuration parameter that was modified was the format of the data being recorded. GNSS-SDR allows user to record the navigation solution using both a text file with the position and velocity, and also a RINEX file. RINEX stands for Receiver Independent Exchange Format and is a common format for raw satellite navigation systems. Additionally, GNSS-SDR was configured to record observables from incoming signals for the tracking of any satellite even without a position fix. This way, data from in orbit can still be utilized to verify hardware components without a complete navigation solution. This data includes data such as signal strength, pseudo-range and Doppler shift. This data is converted and saved as a Matlab file.

3. Configuration Files

Each run of GNSS-SDR requires a .conf file that describes the configurations for the SDR. In this section, the 4 configuration files created for this project will be described in terms of their application and differences. These files can be found in the GitHub repository gnss-sdr-rpi.

1) postProcessed.conf: Decodes GPS signals that have already been prerecorded by an SDR and are stored in a file on the computer running GNSS-SDR. Useful for verifying software. Does not require a peripheral since signals have already been recorded and conditioned for a computer.
2) realTime.conf: Attempts to obtain a navigation solutions using real time signals from the surface of the Earth received by a HackRF. Useful for testing the complete set up but requires user to have a clear line of sight to the sky.
3) rtl.conf: Has the same purpose as realTime.conf but uses the RTL-SDR dongle which can increase the ease of taking data or running quick tests.
4) InOrbit.conf: Is designed to operate in orbit for the HackRF. The main difference with InOrbit is the increase in Doppler range.

4. Deciphering GNSS-SDR Messages

One of the most important tools one has when conducting SDR tests is the terminal output from GNSS-SDR that give insight into the operations of the SDR as it interacts with GPS signals. Thus the following list describes the type of messages received from GNSS-SDR and what they mean.

1) Starting message: This message includes the configuration of GNSS-SDR including the sampling rate, the frequency, the signal source, etc.
2) Current receiver time: This updates every second with the amount of time since the receiver began searching for signals.
3) Tracking of: The tracking of message will alert users what satellite the receiver is attempting to connect with, via PRN, and on which channel. It is import to note that this does not mean that any signals have been received by the SDR. This is helpful for debugging because it allows users to determine if the SDR is failing to interpret signals or if those signals simply are not there. For example, if the user knows that GPS satellite PRN 9 is directly above head and they see a message saying Tracking of GPS PRN 9 that does not lead to a navigation message then users can be assured that the receiver is not working correctly.
4) Bit synchronization: Users will see a bit synchronization when GNSS-SDR has locked onto a signal and began correlating the GPS signal with the signal simulated by GNSS-SDR.
5) New GPS NAV message: This message alerts the user that navigation messages (for example pseudo ranges) are now being received from a specified satellite.
6) Loss of lock: This message appears when GNSS-SDR has searched the expected frequency window for a certain satellite and has not been able to acquire any signals. The message alerts users that the software will begin using that channel to search

for other satellites. It is important to note that this message does not mean that signals were necessarily ever received.

7) Position at: This message will convey the UTC, receiver position (in latitude, longitude and altitude) and receiver velocity (in East, North, up) when a navigation solution has been achieved.

5. Data Management

The SDR subsystem creates several different types of data, each with a different value to the mission. Additionally, this data is created while in orbit meaning the ability to access the data is limited. Thus there is a need to manage SDR data. This was accomplished by Python scripts that run on the CM4 and will be discussed in more detail in the next few paragraphs.

SDR data is prioritized by two metrics: length of experiment and if there was a navigation solution. Length of experiment was prioritized due to the fact that longer experiments mean more data and allow for better understanding of how the system in behaving. Secondly, runs with a position fix are prioritized so they can be used to compare the accuracy of the SDR as a GNSS receiver against the commercial navigation unit that is also on board the satellite. Each of these metrics are checked by a python script that parses through the text output from GNSS-SDR.

Each run of GNSS-SDR will create a folder of data that will include the duration of the experiment and the validity of the navigation solution (yes or no) in the folder name. Within each folder there will be two folders, one meant for serial communications and one meant for data downlink. The data downlink file will include a RINEX file, a complete text log of the GNSS-SDR output, and Matlab files with signal observables like Doppler data. The contents of this file will be down linked through an S-Band radio using libcsp.

In the case that the S-Band radio experiences issues, the system is prepared to send data via a serial connection to the main OBC which can then downlink the data. However, the storage space on the main OBC is less than that of the CM4 so only small chunks of data can be sent. In this situation the CM4 will only send a singular navigation solution for a singular epoch. This is done using PySerial protocol with a predetermined message format including parity bits as to avoid error from a bit flipping.

D. Initial Testing

The challenges of spaceflight demand extensive testing to ensure both capability to survive the harsh environment of space and the ability to operate with minimal commanding. The SDR payload for OrCa2 is no exception, and in this section the preparation for and completion of several important tests will be discussed.

1. Testing with Prerecorded Signals

One of the first tests able to be conducted with the SDR system is verifying that GNSS-SDR was correctly operating through the use of prerecorded signals. The prerecorded signals are GNSS signals that have been processed only by a SDR peripheral, meaning the signals are quantized into packages that can be deciphered by an SDR computer. For testing on this project, the prerecorded GNSS signals were acquired through the GNSS-SDR website. Running this kind of testing allows verifies that GNSS-SDR is running correctly, that the configuration file being used is working, and that the computer has the proper processing speeds to get a navigation solution.

While this is a good start to testing the SDR system, this does not give any insight into the performance of the TCXO, antenna or SDR peripheral. It should be possible to remedy these shortcomings by using a second SDR set up to record and playback raw GNSS signals. A HackRF connected to a laptop could be used to record raw GNSS signals from the ground using a simple recording configuration with GNU radio. Then the HackRF used to record these signals would be connected to the SDR system via coaxial cables to playback these raw signals. Thus giving the SDR system a chance to utilize all components in a honest navigation solution procedure. It should be noted that there are challenges associated with this procedure. For one, the size of GNSS signal recordings can become very large and difficult to manage. A short recording of 30 seconds resulted in a file of several gigabytes which was unable to be opened by the mac book being used. Additionally, one must be very careful to never broadcast these signals over the air since this violates federal law.

Having a prerecorded procedure as previously described would not necessarily provide more information about performance than simply taking the SDR system outside to perform a live navigation solution, having the prerecorded files would allow testing of nominal operations from any position such as while attached to a flat sat or indoors without good view of the sky.

2. Testing with real time signals

The most common type of test ran with the SDR system was real time testing. This involves taking the set up to a location with an unobstructed line of sight to the sky and performing GNSS-SDR with signals incoming from currently in orbit GNSS satellites. For this project, only GPS satellite signals were considered.

This is an important type of test that provided imperative information to the research team. Firstly, this type of test allows verifies the capabilities of all hardware components. Secondly, this test produces data of similar type as would be seen in orbit giving, thus trial data to work with. Thirdly, the data produced by a real time signal test can validate

that the SDR management software is working correctly in terms of storing and prioritizing data.

There are drawbacks with an outdoor test. The conditions are variable and often outside of the control of the researcher. For example, heavy clouds or rainfall would prevent a navigation solution. Additionally, requiring an unobstructed sky reduced the possible locations for testing. The most successful locations were on rooftops and in the middle of large fields, but having access to charging points in these locations can be difficult. And, perhaps most critically, taking sensitive equipment meant for final use on the satellite outdoors can cause damage of the devices and lead to failure in orbit. Thus, real time signal tests were never conducted with any flight hardware, but instead flight spares.

It is worth noting that taking the entire SDR system proposed in this paper outdoors can be challenging due to the many parts, need for monitors and keyboards, and lack of charging points. Thus, most tests were conducted with alternate set ups. For example, often times the CM4 was substituted for by either a laptop or the Raspberry Pi 4 (which has the same processor as the CM4). An image of an example alternate setup used for outdoor real time signal processing is shown in Figure 10. In this setup, the CM4 has been replaced by a personal laptop. In this particular set up, a Beaglebone black, an embedded Linux device, is used simply to convert USB 5 volt power from the laptop to 3.3 volt to power the TXCO.

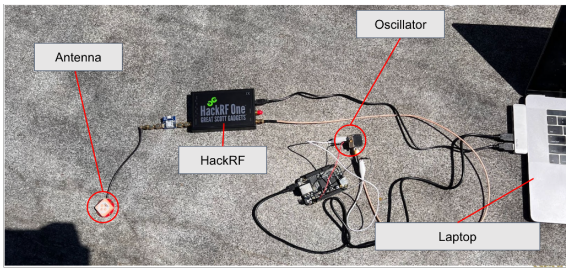

Fig. 10 Testing Setup for Real Time Signal Testing

The real time testing set up was quite successful in terms of obtaining a position fix. On a clear day with minimal obstructions, the real time testing set up obtained a position fix often in less than 5 minutes. Typically with an accuracy within 20 meters and a velocity error less than 4 m/s.

3. Testing on GNSS Simulator

Perhaps the most crucial testing done for this SDR system was done through GNSS Simulator testing. The NavX -NCS Navigation Constellation Simulator was used for GNSS simulation testing. This simulator allows users to define the parameters of the simulation including the GNSS constellation being simulated, the location of the receiver, the dynamics of the receiver, etc. These parameters will be discussed in more detail later in this section.

1) Time: Users can determine what time they wish to simulate the GNSS signals for. If a point in the future is chosen the simulator software will propagate the orbits of each GNSS satellite forward to that time. Since the operations for this SDR system does not have a specified time of operation, this parameter was usually left to just be the present time.
2) Location / Orbit: Users may specify their dynamic model which can be either stationary, dynamics or orbital. Stationary allows users to input a latitude, longitude and altitude, which was used for early tests. The orbital option allows users to input their desired orbit via orbital elements. This allowed the input of the orbit of OrCa2, to evaluate how well the system will perform during the mission.
3) Signal Power: The signal power can be configured to either be a constant value, meaning each signal is broadcast from the simulator at the same decibel level. The signal power can also be configured to transmit level, meaning the signal power loss due to free-space path loss and atmospheric attenuation is accounted for. To account for the gain that is added by antennas, the signal power constant mode was used and set to the values expected in orbit. These values were obtained by a link budget taking into account maximum path loss and antenna gains.
4) Constellation: Users can choose what constellation they are modeling from the options of GPS, BeiDou, Galileo, QZSS, SBAS and GLONASS. For this project only GPS was considered.
5) Data Files: Users can configure the software to produce different types of data. This includes RINEX files for each simulated satellite and text files describing the simulated position of the receiver. The text file describing the simulated position of the receiver was used to assess the accuracy of the SDR system.

There is a laptop connected to the GNSS simulator with software allowing users to configure the GNSS simulator. The parameters that were specified when testing the SDR system are listed and described below.

Working with the GNSS simulator requires an abundance of caution due to the fragility and high price point of the GNSS simulator. However, if proper operations are understood then testing the SDR system is relatively straight forward. The GNSS simulator has an SMA cable that outputs the raw generated GNSS signals that can plug directly into the HackRF. However, it is best practice to place a bias tee in between the GNSS simulator and the HackRF. In this instance, the bias tee is being used to prevent any voltage from being supplied to the GNSS

simulator since this could cause permanent damage to the simulator. Bias tees are often used in GNSS applications to supply voltage to antennas, however we are able to apply voltage to the antenna with the innate capabilities of the HackRF. A diagram of a bias tee circuit is shown below in figure 11.

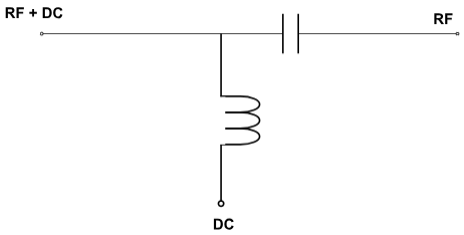

Fig. 11 Bias Tee Diagram

The GNSS simulator also provides several tools during the simulation that can be utilized to increase understanding of the SDR system's behavior. Perhaps the most notable is the SkyPlot that is generated showing which simulated GNSS satellites are in view of the receiver. This is useful in debugging the system, since this can help determine if the receiver is failing to correlate with the GNSS signals or if the signals simply are not there currently.

4. Thermal Vacuum Chamber Test

Another important test to ensure performance in orbit is a thermal vacuum test. The space environment is demanding both due to the large temperature swings seen in orbit and the constant vacuum of space. Thus, it is an important step in hardware preparation to test how equipment designed to fly will withstand these challenges. The research team has access to a thermal vacuum chamber that will allow just this. At the time of this writing, the thermal vacuum test has yet to be conducted, however detailed plans have been developed and will hopefully be enacted within the coming weeks.

E. Performance of SDR as Position Fix Device

The performance of both the HackRF and the RTL-SDR dongle were measured through several means and displayed in Table 2. Firstly was the average time to obtain a position fix. This metric is crucial since the SDR is a power heavy element of the CubeSat and, in most CubeSat applications, can only be powered for short periods. The average time for a position fix was measured by running the most effective GNSS-SDR configuration while testing on the GPS simulator. These values were obtained by averaging over three runs. The time is measured from when GNSS-SDR begins to when the first navigation solution is provided.

The position and velocity error were obtained through the GPS simulator as well by comparing the position and velocity prescribed by the GPS simulator and the position and velocity calculated by the SDR.

The power draw steady and power draw maximum were both observed by assembling the entire SDR payload including the CM4, the HackRF, the TCXO and the antenna (with a bias voltage) while performing a position fix. Through the use of a USB power meter, the power draw of the system was able to be recorded.

Table 2 SDR Performance as a Navigation Unit

Metric	Performance	Units
Time to Fix	10.2	Minutes
Position Error	25.3	meters
Velocity Error	5.4	meters/second
Power Draw Steady	5.10	Watts
Power Draw Maximum	6.2853	Watts

V. Discussion

A. Benefits

When assembled, the SDR system that will fly on OrCa2 boasts a very low price point less than $550, about 10 times less than most commercial navigation units. Additionally, integrating the custom navigation unit into the rest of the satellite was uncomplicated by the open source nature of the software and popularity of the hardware. The CM4 is especially easy to work with, making communication with the main OBC readily manageable. The ease of integration for the SDR system would allow for fast turn around times for projects with short timelines. The position fix provided by the custom navigation unit is both timely and accurate.

B. Limitations

Perhaps the most challenging limitation to the SDR as a GNSS receiver is the power requirements. GNSS receivers from commercial vendors have a typical power consumption less than 2 watts. Any CubeSat mission with long periods without power generation or a need for constant position and velocity awareness would struggle to keep up with the HackRF power demands. The HackRF also has a much larger form factor than traditionally GNSS receivers, being approximately 3.5 larger in both volume and mass.

Additionally, the HackRF could be seen as risky to fly due to its lack of flight heritage, however gaining flight heritage is a major goal of the OrCa2 mission. But speaking more generally, the HackRF is not space rated nor does

the company offer services to make the device space ready. Not only are the commercial GNSS receivers space rated, but they also often provide documentation and insight into how to fully prepare the device for space flight. This will need to be developed for the HackRF over time.

C. Future Work

The reason for which the OrCa2 team was so excited to work with the HackRF is due to its wide range of capabilities. While this research focused on the application of a HackRF as a navigation unit, there is still much open work to be done to understand how the HackRF could provide other onboard services. Perhaps most interesting would be the potential of receiving commands and communicating data while in orbit, either to other satellites or ground stations. The Opera Cake is an add on to the original HackRF one that should be researched for future missions since the Opera Cake board allows for antenna switching. An onboard Opera Cake could facilitate a multitude of modes for a single HackRF.

Once flight heritage on the HackRF is gained and there is confidence about its in orbit performance then there are many paths one could take to make the overall SDR system more modular. If one of the main ideas behind using an off the shelf SDR is fast turn around times then in future iterations modularity should be a key focus. One could make a single PCB to assist with the integration between the CM4, HackRF and oscillator so it could be treated as a single component.

VI. Conclusion: Navigation Unit

The OrCa2 team demonstrated the feasibility of using off the shelf components to create an SDR enabled GNSS receiver. The ability to obtain a navigation solution both while on the ground and with simulated signals representative of what will be seen in orbit was demoonstrated. The SDR enabled system provided an accurate navigation device at a low price point with parts that are comparatively easy to integrate. However, the large form factor, demanding power requirements and lack of flight heritage make prevent the SDR navigation unit from being an objectively preferred alternative to the commercially vended options. Future work that leverages the flexibility of SDRs and increases their flight heritage may make these options more preferred in the not too distant future.

VII. Motivation: Initial Orbit Determination

The satellite that this experiment is being flown on, OrCa2, has a primary objective of increasing space situational awareness (SSA) of resident space objects. Thus, it is fitting and pertinent that the software defined radio experiment can be considered to do just that. While the reflective panels on OrCa2's exterior will better models for monitoring RSOs in the visible spectrum, radio receivers offer the ability to monitor RSOs in the radio spectrum. This would only apply to space objects that are actively transmitting, which is a large percentage of objects currently in orbit. Thus, for this project the on board SDR was considered as a potential external orbit determination device, as in determining the orbit of other satellites. Having an on board external orbit determination device could not only increase overall SSA but could create an easy on orbit device to help with collision avoidance or perform reconnaissance. In this part of the paper, the authors will discuss the potential ways to perform external orbit determination, then will focus on the use of Doppler Only IOD by showcasing simulated and field results.

VIII. Methodology: Initial Orbit Determination

A. Measurables

It might seem counter intuitive that a single radio could perform external orbit determination without the need to decode or understand the message within the transmission. However, there are two main metrics that can be determined through just signal reception: Doppler shift and Angle on arrival (AOA)

1. Doppler Shift

The Doppler shift, or Doppler effect, is a physical phenomenon the describes both electromagnetic and physical waves that have a velocity relative to the receiver. A transmitter will create waves that will propagate at a constant speed, however if the transmitter is moving in the direction of the transmission, the waves will begin to bunch up. This will decrease the wavelength and thus increase the frequency. In a sound wave, this can be observed as a change of pitch. In a radio wave, this can be observed as a change between the transmit frequency and the received frequency, referred to as the Doppler shift. This phenomena can be visualized in Figure 12.

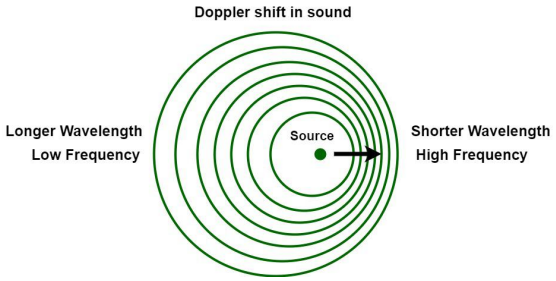

Fig. 12 Visualization of the Doppler Shift [5]

With this information in mind, if one knows the transmit frequency and the observed frequency then they can deduced the expected relative velocity. The Doppler shift can then be applied to transmitting satellites to determine their relative velocity to the observer. This relative velocity can be thought of as the range rate, or how quickly the distance from the observer to the satellite is either increasing or decreasing.

In order to determine the relative velocity however one must know the transmit frequency which is not always the case for active satellites. One idea to remedy this would be to search in the commonly used frequency bands (L1, S, K) and looks for the best match between the predicted and received Doppler curves. Another, potentially more practical, idea is to utilize the unique geometry, figure 13, of an overhead pass to obtain the transmit frequency. This concept relies on the fact that when a satellite passes overhead it will have to switch from moving closer to the observer to then moving farther away. This means that at a single instance of time the range rate is zero therefor the received frequency and the transmitted frequency should be the same.

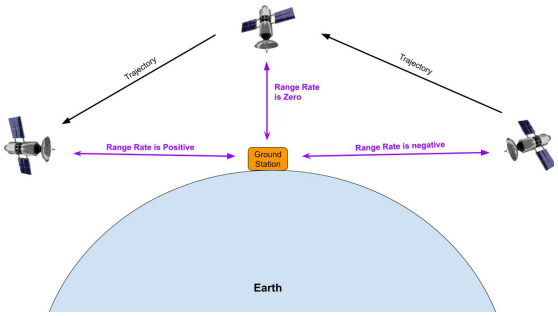

Fig. 13 Change in range rate during an overhead pass

Since the observer would have no inclination as to when the overhead pass is taking place, this timing can be determined by looking at the frequency on arrival graph. An example graph has been generated and can be viewed in figure 14. This graph is displaying an overhead pass of a satellite transmitting at L1 (1575.42 MHz). As can be seen, in the time leading up to the overhead pass, the received frequency is above the transmitted frequency and after the overhead pass it is lower. Thus, the point of transfer from high to low frequency, or the inflection point, of this graph will show when the satellite is overhead. The inflection point is found by fitting a polynomial to the data using the Matlab fit function and finding the zero points of the double time derivative.

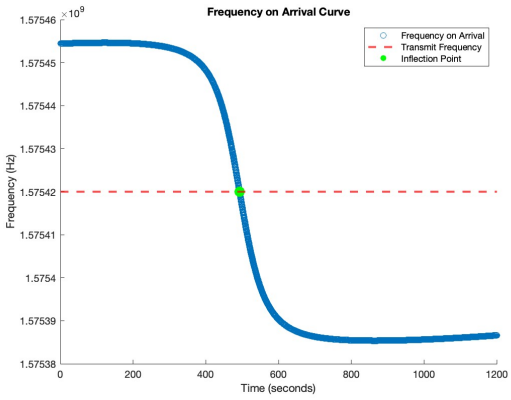

Fig. 14 Frequency on arrival curve for overhead pass

2. Angle on Arrival

Another approach to determining the orbit of a transmitting satellite from an SDR, is to look at the angle of the incoming signal, or angle of arrival (AOA). The AOA method consists of using an antenna array to determine the angle between the incident signal and antenna . To better understand how this concept works please refer to figure 15. In the figure, A1 and A2 represent two distinct antenna at a finite distance apart, Δx. If the signal source is significantly far away, like they are when looking at transmitting satellites from the ground, then the incoming signals lines can be considered parallel.

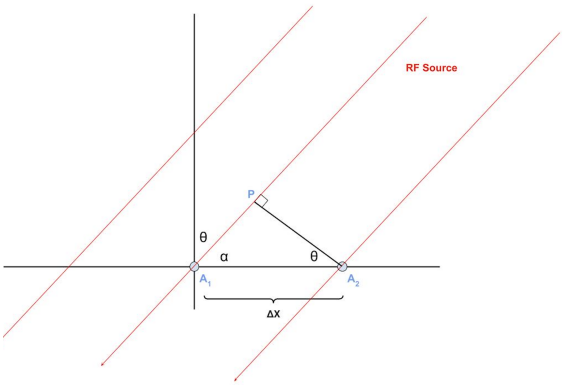

Fig. 15 Angle of Arrival using antenna array

Theta is defined as the angle between the antenna array normal and the incoming signal. The distance between point P and A_1 represents a finite distance that must be traveled by the signal to reach A_1 after the signal has been received by A_2. When the signal propagates this additional distance, there will be a change in phase. This change is phase is given by equation 1 in which λ is the wavelength of the incoming signal.

$$\Delta\phi = 2\pi(\frac{\Delta x * sin\theta}{\lambda}) \quad (1)$$

Equation 1 can be rewritten to solve for theta as in equation 2. With this knowledge, an antenna array with the ability to measure and compare phase of the incoming signal can determine the angle of arrival. For more information on how to obtain an angle on arrival measurement see reference [3]

$$\theta = sin^{-1}[\frac{\lambda(\frac{\Delta\phi}{2\pi})}{\Delta x}] \quad (2)$$

This method has been tested with SDRs and specifically with a HackRF. Reference [2] implemented an array of multiple HackRFs to estimate the signal direction and were able to do so with an error of 1.7°. However, since the HackRF is already a power heavy device, it may be unreasonable for most CubeSat missions to consider flying more than one.

B. Doppler only IOD

The Doppler only initial orbit determination (DO-IOD) method was pioneered in 2022 by John Christian, Christopher Ertl, Kenneth Horneman and T. Alan Lovell. The intention for DO-IOD was to not only be able to correct apriori guesses with Doppler data but to be able to obtain the orbital parameters of a transmitting satellite from Doppler data alone.

To better understand how DO-IOD works please see reference [4], however the following paragraphs will given an overview of the basics behind DO-IOD. Note that DO-IOD does require for the orbit of the satellite to be close to circular. The potential ranges of eccentricity will be explored later in this paper. However, by constraining the DO-IOD problem to only circular orbits, this limits the IOD problem to a single parameter search.

Start with the simple idea that the range between a satellite and a ground station is given by equation 3 in which ρ is the range, and subscripts GS and S denote the ground station and the satellite respectively. Equation 3 can be written more succinctly if r_S and r_{GS} represent the position vector of each point as in equation 4. The time derivative of equation 4 is given by equation 5 in which $\dot{\rho}$ represents the range rate.

$$\rho = \sqrt{(x_{GS} - x_S)^2 + (y_{GS} - y_S)^2 + (z_{GS} - z_S)^2} \quad (3)$$

$$\rho = \|r_{GS} - r_S\| \quad (4)$$

$$\rho_i\dot{\rho} = (\mathbf{v}_i - \mathbf{v}_{S_i})^T(\mathbf{r}_i - \mathbf{r}_{S_i}) \quad (5)$$

The range rate, $\dot{\rho}$, can be related to the received frequencies through the Doppler shift. The Doppler shift is defined by equation 6 and allows users who know the original transmit frequency to determine the range rate between observer and transmitter.

$$f_i = \left(1 - \frac{\dot{\rho}_i}{c}\right) f_0 \quad (6)$$

Utilizing the assumption that the transmitting satellite is on a nearly circular orbit, the velocity and position of the satellite can be related using equation 7. By using f and g functions the initial condition can be mapped forward in time.

$$v_i^2 = \frac{\mu}{r_i} \quad (7)$$

This results in two n by 1 matrices equated to one another, where n is the number of measurements. One matrix describes the range rate as a function of Doppler values and the other in terms of position and velocity. However, for a circular orbit with the f and g formulation the position and velocity is simply given by equations 8 and 9 in which p and q are the axes in a perifocal frame. Thus there is only a dependence on r, the semi-major axis of the orbit. This means that the system can be solved in a least squares sense.

$$r_0 = r\boldsymbol{p} \quad (8)$$

$$v_0 = \sqrt{\frac{\mu}{r}}\boldsymbol{q} \quad (9)$$

For a more detailed derivation and information on the estimation process used for DO-IOD please see reference [4].

C. Angles only IOD

As mentioned earlier, it is possible to measure the angle to a transmitting satellite via an antenna array with an SDR. To fully know the state of an orbiting object one must know 6 independent elements, thus knowing the angle to a satellite, which is only two independent measurements, does not suffice to determine the orbit. However, taking multiple measurements of angles, at least three, can result is a solvable system.

The study of orbit determination via angles alone has long been a topic of interest, and was relied upon heavily in the history of astronomy. In 1780, Laplace suggested a method by which with 3 pairs of angles from a topocentric frame one can determine the orbit of an Earth satellite. The derivation of this method will not be discusses here but can be read in Section 2.11 of reference [1].

D. Utilizing angles and Doppler

One method, that will not be explored experimentally in this paper ,would be the idea of combining the knowledge of the angle to the transmitting satellite with the Doppler data. Both types of measurements can be used to produce an initial state estimation, however DO-IOD assumes a very low eccentricity for the transmitter. Thus, it would be wise to begin with an angles only method to guess the initial state and the eccentricity. Then a batch estimator could be employed that takes into account both the angles data and the Doppler data to attempt to correct the initial guess. However, the weighting of the Doppler measurements should be proportionally scaled with the inverse of the eccentricity. As in a highly elliptical orbit should rely solely on the angle measurements but a nearly circular orbit should rely on both Doppler and angle measrurements more equally.

E. Simulation testing

In order to investigate Doppler-Only IOD, a simulation environment in Matlab was generated. This environment allowed validity testing of the IOD algorithm, vary input variables and provide data to check the real time testing against. The general structure of the simulation environment is displayed in figure 16. Having the simulation environment also allows one to study how different starting conditions may affect performance. In this section, an analysis of both the eccentricity and frequency noise limits of DO-IOD will be shown. Then the Doppler only IOD algorithm will be tested for 3 cases: Ground pass of LEO satellite, ground pass of GPS satellite, and orbital pass of GPS satellite.

The runs of the different simulations will all be evaluated on the uncertainty in position and velocity for the starting epoch. Additionally, how closely matched the simulated and predicted frequency on arrival (FOA) curves are will be analyzed. The simulation environment propagates satellites forward in time from a starting set of initial conditions, either orbital elements or a position/velocity , using Kepler's method.

1. Investigating Transmitter Eccentricity

One of the base assumptions within DO-IOD is that the orbit is circular. Of course, no orbit is perfectly circular meaning that there is a certain degree of tolerance with the orbits eccentricity. Thus, a better understand of the range of acceptable eccentricities was sought out, in order to better understand the potential applications of the DO-IOD method.

In order to isolate the impact of eccentricity, a very simplified scenario was constructed. In the scenario the receiver is on the surface of the Earth and experiences no random noise on the measurements. Additionally, for simplicity, it was assumed that there was constant line of sight between the ground station and the satellite, ignoring the signal blockage caused by the Earth. While this assumption deviates greatly from the real world, it made for easier debugging with the eccentricity investigation.

The simulation runs for 1 period of the transmitting satellite and takes 15,000 samples at even intervals. A LEO orbital satellite, with an altitude of 441.44 km, was chosen for this investigation.

2. Investigating Impact of Noise

With the simulation data, if executed correctly, one can simulate the exact measurables of the scenario. While this is good for simulation validation and research into the algorithm itself, it is very unrealistic. Thus, to simulate the noise seen by actual instruments, a random error was added to each frequency measurement. The user can define a term deemed "scaling factor", and an error up to the magnitude of the scaling factor will be added to each frequency measurement. For example, a scaling factor of 1 Hz will lead to a random value between -1 and 1 being added to each measurement value.

In order to better understand the DO-IOD algorithm and its noise limitations, a simplified scenario was created, in which there were multiple runs with the only variable being noise level. In order to isolate the impact of noise, a very simplified scenario, identical to the one used for the eccentricity investigation, was constructed. In the scenario the receiver is on the surface of the Earth and experiences no random noise on the measurements. This includes constant line of sight from a ground station, 15,000 samples taken and a LEO satellite with a 441.44 km altitude.

3. Investigating Ground Pass of LEO satellite

The Doppler only IOD method has already been tested for the case of a LEO satellite. However, in this research and this paper the methodology and results from this case can be seen as a baseline for the other experiments. The ground pass scenario that will be modeled will be the pass of a LEO satellite with inclination of 35.37°, an eccentricity of 0.0013, and an altitude of 441.44 km .

4. Investigating Ground Pass of GPS satellite

The change from a LEO satellite to a GPS satellite was done as one step towards simulating the in space pass of a GPS satellite. Changing to a GPS satellites involves an increase in semi major axis by a factor of approximately 4 and an increase in eccentricity of about 6. Similarly to the LEO satellite, the GPS satellite was simulated for 3 passes of the ground station i.e. 3 segments in which there was a valid LOS between the ground station and the satellite.

The state of the GPS satellite was obtained using the

gnssconstellation function in the navigation toolbox for Matlab. GPS satellite with PRN of 9 was chosen for the simulation.

5. In Orbit pass of GPS satellite

This model is meant to represent the actual circumstance seen by an orbiting satellite. The receiver will no longer be a stationary point on the Earth, but instead be a point moving along the trajectory prescribed by user inputted orbital elements. To simplify the model, it is assumed that the receiving spacecraft can receive radio signals regardless of the orientation of both satellites. This simulation was run for one full orbit of the transmitting GPS satellite.

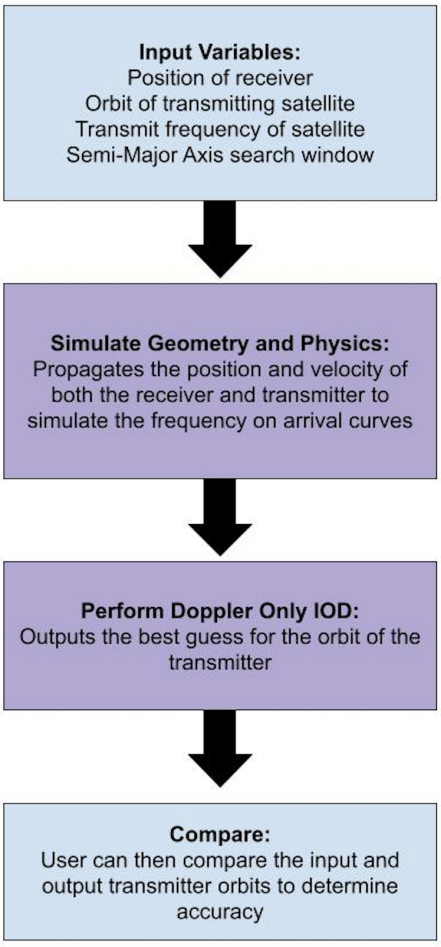

Fig. 16 Structure of Simulation Environment

F. Real time testing

The ultimate goal of this research is to perform DO-IOD with real time data obtained by an SDR. In order to test this, HackRF with a laptop was used to get live Doppler curves from a GPS satellite. Utilizing the GNSS-SDR software, Doppler curves for any overhead GPS satellite, and the associated PRN for the satellite, were obtained. Measurements were taken in the middle of a large field on Georgia Tech's campus with SDR equipment including a magnetic L1 antenna.

In order to apply DO-IOD to the recorded data, users need to determine a few other parameters, namely the location of the receiver and the time of the satellite pass. Both were obtained through the position fix provided by GNSS-SDR. When the SDR system gets a navigation solution it will print out the latitude, longitude and altitude of the receiver and the associated UTC time. These parameters, along with the Doppler curve, were then put into the DO-IOD method to attempt to produce a convergence onto the GPS satellite orbit.

IX. Results

A. Eccentricity Limits

The results from the testing of eccentricity are plotted below. Figure 17 shows how the position error of the initial guess when compared to the truth increases with eccentricity and figure 18 shows how velocity error changes with eccentricity.

It is clear from the figures that an eccentricity greater than 0.05 does not lead to an accurate solution. The first two trials of 0 and 0.01 produced data that converged very well and produced solutions within a meter of the truth position. Thus until further investigations are conducted, or the DO-IOD algorithm is modified, users should only apply this method to satellites with an eccentricity less than 0.01. This means DO-IOD can still be applied to many Earth satellites, including GPS.

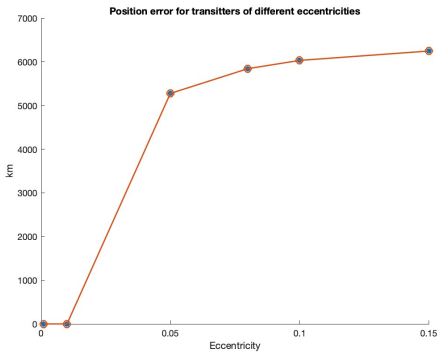

Fig. 17 Position Error for Transmitter with different Eccentricities

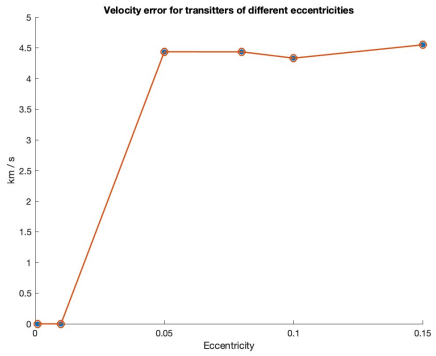

Fig. 18 Velocity Error for Transmitter with different Eccentricities

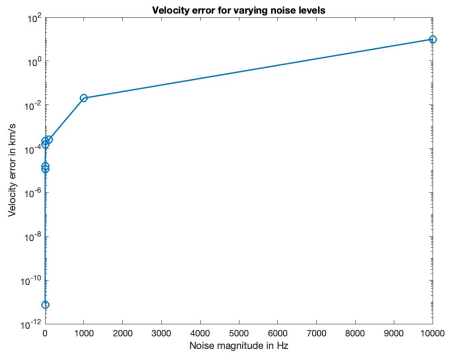

Fig. 20 Velocity Error for varying noise levels

B. Investigation on frequency noise

The position and velocity error for different noise levels are plotted in figure 19 and 20 below. From the plots it can be seen that for the idealized scenario, the DO-IOD algorithm still converges for frequency noise less than 1000 Hz. However, for orders of magnitude larger than 1000, the solution quickly diverges. For the noise level of 10,000 the position estimate is off by more than 10,000 km.

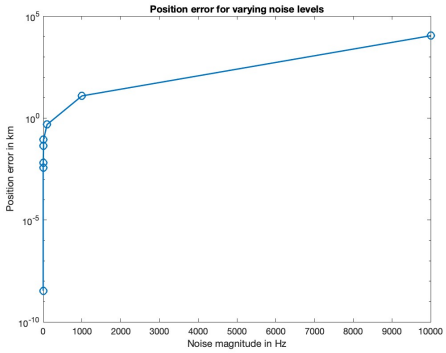

Fig. 19 Position Error for varying noise levels

C. Simulated Passes

1. LEO ground pass

For the LEO ground pass the Doppler Only IOD algorithm produced a solution with minimal error. The initial conditions predicted by the algorithm are off from the truth by 4.7753e-04 meters and 2.7466e-7 meters per second. The simulation successfully produced similar results as to those from the real time LEO pass from reference [4].

To understand how accurate this solution is, two satellites were simulated, one with the actual initial conditions and one with those predicted by DO-IOD. The difference between the two orbits are plotted in figure 21 where the y axis represents difference in position in kilometers. As can be seen in the plot, the small discrepancy in initial conditions results in only a few meters of difference over the course of a month. However, this is ignoring drag, solar radiation pressure and any other type of perturbing force.

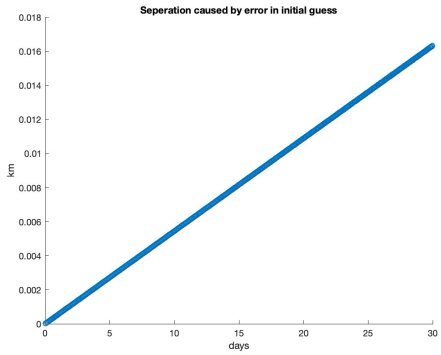

Fig. 21 Propogation of position error from inital guess

2. GPS ground pass

For the GPS ground pass the Doppler Only IOD algorithm was able to obtain the initial state of GPS 31 with a position discrepancy of 4.6927e-03 meters and a velocity offset of 7.1689e-7 m/s. Figure 22 shows the simulated FOA values and those predicted by the solution to the Doppler POD method. As you can see the two curves are perfectly overlaid, to the point where the simulated values can not be seen.

In order to get a better understanding of the type of results one should expect to see with actual instruments this simulation was run again but increasing the noise from 0 to 10 Hz. This resulted in the plots shown in Figure 23 and the following uncertainties: Position error of 2.3425e+04 km and velocity error of 2.3091 km/s.

These results indicate that if the real time testing from the ground has any noise then a convergence on to a solution

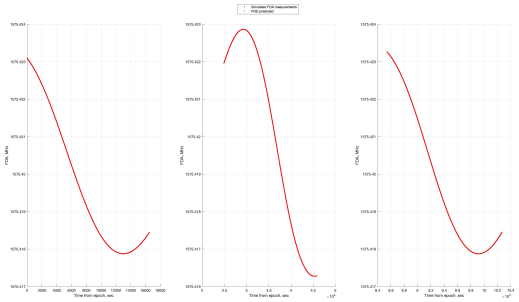

Fig. 22 FOA for three passes by GPS satellite of singular Ground Station

might not be possible with the current state of the estimation code.

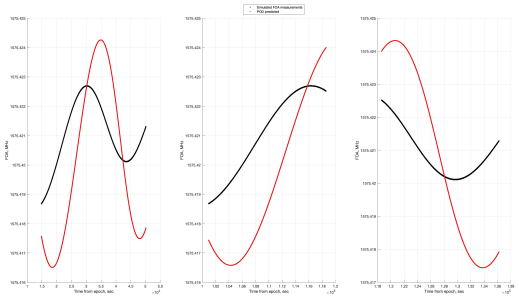

Fig. 23 FOA: GPS satellite ground pass with 10Hz of noise

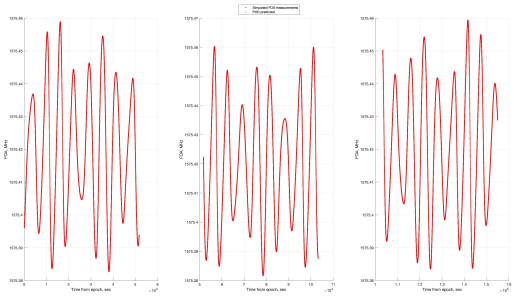

Fig. 24 Simulated and predicted FOA curves for orbital pass

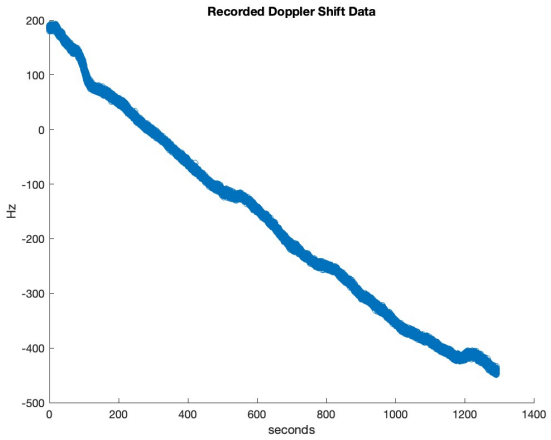

Fig. 25 Recorded Doppler Shift from GPS

3. In Orbit Pass

The in orbit pass was able to produce an adequately accurate solution with a position uncertainty of only 65 m and velocity error 1.0572e-02 meters per second. The orbital pass provides unique data since there are more often passes as compared to when there is a stationary observer. Figure 24 shows the predicted and simulated FOA curves for the 12 hour period in which this scenario was simulated.

D. Real time testing

The recorded Doppler shift from GPS satellite with PRN of 8 can be seen in Figure 25.

The results of this Doppler curve being put through the DO-IOD method was a estimated initial condition that was 3.32e4 km and 4.4 km/s different than the truth. The FOA curves for the measured data and the predicted data are shown in figure 26. The curves have no overlap and show a failure to converge to a solution. The Doppler algorithm actually crashes with this data set and is only able to provide an IOD solution and not a POD solution. The issue stems from the fact that estimated orbital parameters to match the data result in eccentricity greater than 1 which leads to the failure of the POD method.

The exact origin of this eccentricity issue is currently unknown, but it is speculated that the data obtained from the SDR is not truth. After some online queries, it seems as though other HackRF users have experienced an offset in the Doppler measurements. There were attempts to correct the measured data according to the offsets others had measured but to no avail.

It should be noted that, for reasons unbeknownst to the authors, if the receiver was simulated to have an orbit with a non zero inclination the performance of the IOD method was much worse.

X. Discussion: Initial Orbit Determination

A. Benefits

A framework has been created, by which the Doppler Only Initial Orbit Determination method can be implemented on simulated and real data. The framework allows a user to modify several parameters to account for an array

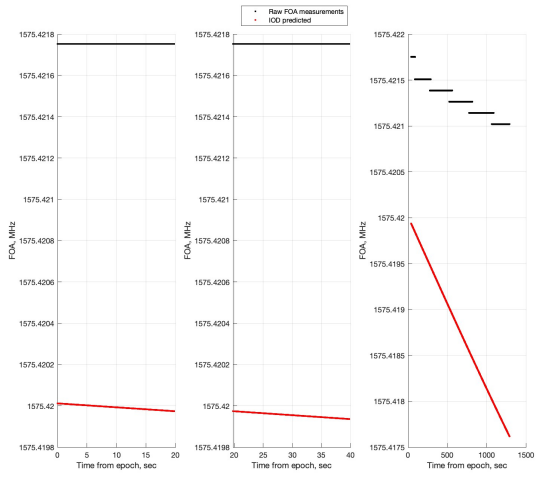

Fig. 26 FOA curves for measured and predicted data

of possible scenarios. These parameters include noise level, position of receiver, orbital parameters of both transmitter and receiver, and others.

The framework was able to demonstrate successful IOD with the DO-IOD methods for simulated scenarios including LEO ground pass, GPS ground pass and GPS in orbit pass. Authors were also able to begin the investigation into the limitations of the Doppler-IOD method looking at the potential ranges for noise and eccentricity.

B. Limitations

1. Simulations

The simulations used in this research can provide insight into the requirements of the Doppler IOD method and the feasibility of applications of this method. However, there needs to be much more testing to understand the nature of the algorithm to better realize the optimal way to configure a scenario. The simulations currently can appear inconsistent, with a small change in the initial conditions leading to a significantly different results.

2. Real Time Data

As of the time this paper is being written, there is still not a means by which an orbit can be initially determined from SDR provided Doppler curves. The recorded data is unverified and even if it were, the complicated relationship between noise and accuracy shown via simulations may prevent an accurate estimation until the DO-IOD code is made more robust.

C. Future Work

The future of this work is exciting! The simulations created through this project have shown the feasibility of using Doppler curves as a means of performing orbit determination on other satellites while in orbit. There are some key steps to take in the future to push for the use of Doppler IOD using the data from the SDR.

1. Doppler IOD robustness

The simulations have exposed several initial conditions that lead to a poor orbit determination solution, that do not make intuitive sense. Thus, it is important that these cases are studied to identify where the problem lies. A great starting point would be to determine the reason why having a receiver in an inclined orbit produces a poor solution.

Additionally, research should be conducted to better understand the relationship between orbit determination accuracy and the number of samples taken. One could conduct studies on the idealized scenario with varying numbers of samples to determine what the preferred sampling rate should be in order to minimize computing time but still provide accurate convergence.

The current state of adding uncertainty to the simulation is adding noise to the frequency measurements but future researchers could tweak other metrics to determine how that effects the overall solution. Perhaps the ground station latitude and longitude or the UTC time of the satellite pass should be slightly deviated from truth.

2. Real Time Data

The first step towards performing Doppler IOD with the recorded data is to confirm that the recorded data is indeed correct. The method for this would be to find the orbital elements for a GPS receiver and simulate the pass to determine how the simulated data and observed data are different and potentially scaling the observed data to match. Perhaps, instead of relying on real time data, one could utilize the GNSS simulator to produce the data to run DO-IOD. This would allow one to know exactly the state of the transmitting satellite.

Additionally, tests should be run on the SDR collecting the data to determine what the average noise is on the frequencies measured by the receiver. One could potentially connect two HackRFs and transmit at the L1 frequency via a coaxial cable. Thus, allowing the second HackRF to record said signals which could be used to determine the average level of noise.

If these steps can be accomplished then a position fix with real time data seems reasonable. This could then be done with the GPS simulator to ensure that the in orbit case is a valid one. Future researchers could also investigate satellites other than GPS satellites with a known ID. For example, if a research satellite was attempting to conduct

Doppler IOD on an unknown transmitting satellite, would users be able to determine the transmitters orbit in one pass? And if not, how does the team identify the same unknown satellite at a later pass?

XI. Conclusion: Initial Orbit Determination

In conclusion, there are several ways to utilize the SDR measured values of angle on arrival and Doppler shift to theoretically determine the orbit of other satellites. The Doppler only method is particularly interesting and was shown, through simulation, to be a feasible option for an in orbit SDR. However, the fragile nature of the algorithm and the lack of reference data prevented orbit determination on SDR recorded data. With further research and better understanding of best methods of handling the batch estimator, there is potential to successfully perform Doppler only initial orbit determination on SDR provided data for any nearly circular orbiting transmitter.

Acknowledgments

Thank you to the Georgia Tech Systems Design Laboratory for providing me with the knowledge and facilities to complete this project. Thank you to Dr. John Christian, who's research in the field of Doppler only initial orbit determination was the inspiration for much of this research. Thank you to Dr. Lightsey, for use of his laboratory's GNSS simulator. Thank you to Dr. Brian Gunter for advising this research and the entire OrCa2 mission. Thank you to Stef Crum, for his advisement on estimation schemes. Finally, thank you to the entire OrCa2 team for aiding in this research, the launch of OrCa2 will be an exciting day.

References

[1] Bate, R. R., Mueller, D. D., and White, J. E., “Further Development and Applications of Stiffness Method,” Fundamentals of Astrodynamics, 1st ed., Dover Publications, New York, 1971, pp. 109–123.

[2] Zihan Zhang, Wenjie Zhao, Yonghui Huang, Junshe An, Yan Zhu, "Implementation of DOA Estimation System Based on HackRF One", International Journal of Antennas and Propagation, vol. 2022, Article ID 7901714, 12 pages, 2022. https://doi.org/10.1155/2022/7901714

[3]Don, M.L., , Wessel, D., Software-Defined Radio Angle of Arrival Measurement of Modulated Signals. 2018

[4] Christian, J., Ertl, C., Horneman, K., and Lovell, A., “Doppler-only initial orbit determination for an orbiting transmitter,” AIAA SCITECH 2022 Forum, 2022.

[5] “Difference between doppler effect and Doppler shift,” *GeeksforGeeks* Available: https://www.geeksforgeeks.org/difference-between-doppler-effect-and-doppler-shift/.