Group Project

VHF Satellite Ground Station

Sławomir Figiel Tomasz Mrugalski Ewelina Omernik

Space and Satellite Technologies

Supervisor: prof. M. Moszyński, Ph.D D.Sc.

Technical supervisor: W. Siwicki, Ph.D.

2020-01-23

GDAŃSK UNIVERSITY OF TECHNOLOGY

Table of Contents

Preamble

Since the initial launch of Sputnik 1 in 1959, thousands of satellites have been launched with a wide variety of tasks to be conducted. Once limited only to military and high-end research organizations with large budgets, with the advent of miniaturization and lower electronic equipment costs, space became more and more accessible to average people. Some space borne equipment, such as satellites or even International Space Station, contains transmitters that are oriented towards amateurs. A good example is a series of Earth Observation satellites launched and operated by the National Oceanic and Atmospheric Administration, a branch of the US government. All of their currently operational satellites – NOAA-15, NOAA-18 and NOAA-19 – contain radio transmitters that broadcast the images of the current state of the atmosphere. The signal is transmitted in the VHF band, with a well-known modulation and data encoding.

1 Project overview

This is a group project being developed as part of the Space and Satellite Technologies studies held at ETI Faculty of Gdańsk University of Technology.

1.1 Project goal

The overall goal of this project is to design, build and then operate a satellite ground station that will be able to receive VHF transmissions from satellites, such as NOAA-15, and then make the data available to a wider community. This will require choosing the appropriate hardware, identifying location for ground station installation, solving logistics regarding power and connectivity, hardware for radio reception and processing, operating the ground station manually, automating many aspects of the ground station operation and then making the resulting data available.

1.2 Project participants

There are several participants involved in the project. **Tomasz Mrugalski** (TM) is a project lead, orbital mechanics specialist, logistics, and a reliability engineer. **Sławomir Figiel** (SF) is a geospatial data engineer, programmer, Raspberry Pi, and an OS specialist. **Ewelina Omernik** (EO) is a low-level software developer, integrated circuits specialist. Prof. **Marek Moszyński**, Ph.D D.Sc is a project supervisor. **Wojciech Siwicki**, Ph.D is a technical supervisor.

1.3 Project Schedule

The overall project time boundaries are limited by the winter 2019/2020 semester terms. The detailed schedule has been proposed and after several iterations agreed on with all major participants. The current schedule is presented in *Table 1: Project schedule* below.

No.	Task	Deadline	Coordinator
1	**Feasibility study** Research of available satellites, capabilities of existing SDR hardware, necessary SDR, antenna and LNA capabilities.	2019-10-17	SF
2	**Hardware acquisition** Selection of specific hardware type, vendor selection, purchasing process, shipment, hardware delivery.	2019-11-07	TM
3	**System integration** Hardware (computing unit, SDR, antenna, wiring), assembly, base software installation (OS, SDR drivers, SDR software)	2019-11-14	EO
4	**Software automation design** Design of the automated data acquisition, processing pipeline, data deployment	2019-11-21	SF
5	**Software implementation** Implementation of the design specified in task #4, developed software deployment	2019-12-19	TM
6	**Test campaign** Test specification, experimental assessment of the system performance, test report, improvement suggestions, conclusions	2020-01-09	EO

***Table 1**: Project schedule*

1.4 Project organization and code repository

A Gitlab instance has been set up at https://gitlab.klub.com.pl:30000/astro/satnog-gdn to streamline the work, keeping tasks and manage the source code. The Gitlab software was designed to manage software projects and it offers many useful features suitable for a project such as this one. A git repository for the source code brings all the benefits of git (version control, change tracking, accountability, history, etc.). Another great feature of gitlab is a powerful issue tracking that offers discussions, easily formulated task lists, content (e.g. images) uploading, easy cross-references and more. Technical discussions are held on gitlab. Many sections of this report reference to gitlab issues (e.g. gitlab #5). To see specific issue, go to https://gitlab.klub.com.pl:30000/astro/satnog-gdn/issues and find the issue number. Note the issue may be closed already. You can go to specific issue directly by specifying it in the URL, e.g. https://gitlab.klub.com.pl:30000/astro/satnog-gdn/issues/5.

1.5 Existing projects

There are several existing projects that provide similar capabilities as planned. The most notable one is Satellite Network of Ground Stations, or SATNOGS [1]. The project aims at providing a common platform for varied types of ground stations with the emphasis on the code being open source. The Satnogs solution does not provide one specific hardware or software recommendation, but instead has a wide variety of options. For example, there are around 6 different antennas mentioned on their wiki page. Pros and cons of each are discussed, but there is no clear consensus which one to be used.

Satnogs provide also a very capable service to store observation artifacts – the decoded images, audio recording, satellite predictions, fly-over diagram, extra comments and more. The database provided has several interesting features. Its primary service is to inspecting the results of the last fly-over. It also has a substantial collection of satellites, with information about their transceivers. The transceivers database proved to be very useful.

However, the Satnogs project has a number of characteristics that makes it less appealing as a student project. First, there is an expectation to use their Raspberry Pi image that has everything included. This would not allow our team to learn great many things. Second, it uploaded the data to a remote services, without any reasonable way to opt out from that upload. Third, our team wants to continue developing the project after the formal time period dedicated to the group project class is complete. Had we based the solution on reacy RPi project, it would be difficult and error prone to participate. This is even more true had we decided to sync with their latest version. As a result, we determined quickly that developing our own software will be more convenient.

2 Ground station development process

This section describes the process that ultimately led to the creation of fully functional VHF ground station that is able to receive satellite transmissions automatically.

2.1 Feasibility study

The first task conducted was a determination whether the data reception from satellites is feasible by a group of students with modest budget. The key concern was whether the hardware required to reliably and repeatedly receive transmissions would be within our budget. Several existing projects were identified with reported repeated successes [1], [2], [3]. The typical radio hardware used was an inexpensive SDR (software defined radio) running on a PC, connected to VHF antenna. In some projects additional components, such as LNA (low noise amplifier) or more advanced directional antenna with tracking mechanism, were used.

Our team looked at various embedded computing platforms. The leading solution available on the market is a Raspberry Pi. Its popularity comes from several factors – affordability (cost around 50-70 EUR), high performance (1.5GHz CPU, comparable to mid-level laptops), availability (sold by many vendors, hardware available in many stores, including those in

Poland), and extensibility: 4 USB sockets for data, powered over USB, Ethernet, some models have PoE (Power-over-Ethernet), some have WiFi integrated, GPIO, HDMI output. Our earliest experiments used Raspberry Pi 1B+ model, which is 5 years old. While it has proven the general approach, it was difficult to work with due to low performance. A decision has been made to migrate to the latest RPi 4B model. For more details, see Gitlab #4.

Another researched aspect was the radio bands. Two most popular bands are VHF and UHF. We decided to use VHF due to being used by several Polish satellites, available antennas and other factors. For more details, see Gitlab #2.

The third researched problem was the choice of antenna. We had to balance several factors here. First concern was he antenna availability. Since the project has strict deadlines imposed, we wanted to get the antenna as soon as possible. Second, the antenna should be reasonably simple to construct. The final aspect was financial. There are many high performance antennas, but their price is often prohibitive. Two final candidates were Winkler turnstile antenna and WiMo TA-1. The latter was slightly more expensive (90EUR, compared to 40EUR), but offered much better delivery options (shipment within 4 days rather than 28 working days).

The deliverable for this task is an analysis with a set of specific hardware selected for purchase. The status of this task is **complete.** The engineer responsible for this task is **Sławomir Figiel.**

2.2 Hardware acquisition

The second task conducted after the feasibility study (see task #1) was to analyze the market from the perspective of available components. Our team looked at several vendors offering different Raspberry PI models via varied channels. Our process covered purchase of three elements: embedded computing platform, a Software Defined Radio component and an antenna.

Embedded computing platform. As determined in task #1 (see the text above), our platform choice was Raspberry Pi 4. It's a very recent model with many powerful features. Our research uncovered stories of users complaining about RPi 4 stability. It seems the problem was faulty design of the USB used to power the solution. This was promptly fixed in an updated 4B versions. The RPi 4B comes with 1, 2 and 4GB memory variants. Since the price difference between the models is not that great, we chose the most powerful model with 4GB of memory. Our rationale for this decision is to be able to run GUI software, such as GQRX, GNU Radio or gpredict on this configured RPi 4B. We also chose a kit that provided several essential hardware. The kit included the board itself, a robust case, micro-HDMI to HDMI connector, a USB-C power fully that can meet the power requirements (constant 3A, even under heavy load), a new micro SD card, and a reader for SD cards. The kit has been purchased on Allegro, a popular sales platform in Poland, on Oct. 22nd and delivered on Oct. 28th. The content of the package is presented in *Figure 1* below.

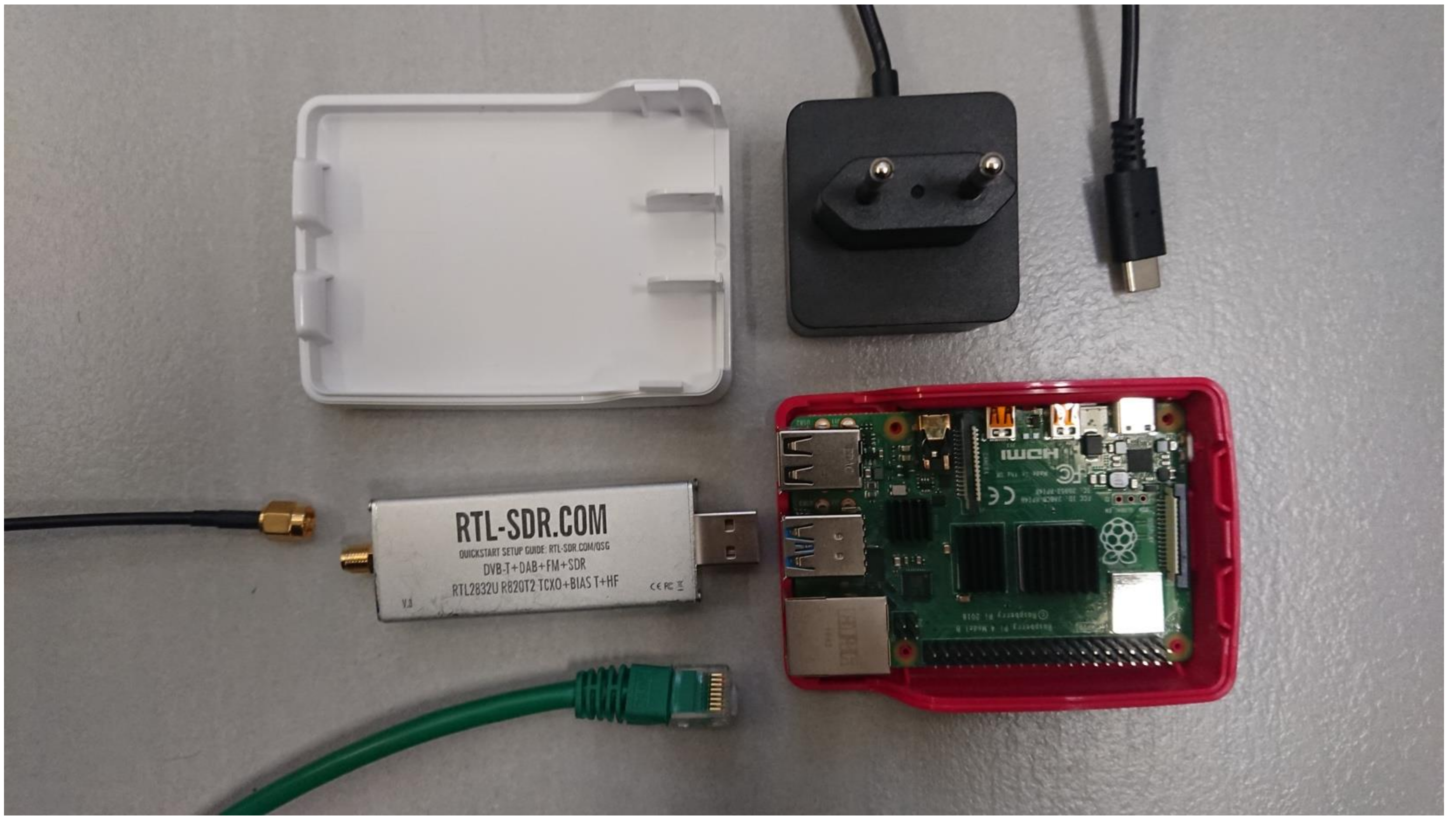

RTL-SDR.COM
QUICKSTART SETUP GUIDE: RTL-SDR.COM/QSG
DVB-T+DAB+FM+SDR
RTL2832U R820T2 TCXO+BIAS T+HF
HDMI

Figure 1*:Ground station components: SDR module, Raspberry Pi motherboard, power supply, case, antenna cable, Ethernet cable*

SDR platform. As determined in task #1, we decided to purchase an SDR platform. By far, the most popular solution is based on two chipsets: RTL2832U + R820T2. Obviously, we needed to connect the SDR dongle to the computing platforms, so it must use USB connector. The model we chose also had a robust case, which protected the delicate hardware inside. The kit we chose came up with a telescope antenna, an SMA cable and a mini-tripod. While we understood the kit antenna is of poor quality, we decided to pay that little extra money to get it, so we could start doing experiments earlier, before the main antenna becomes available. The SDR kit was ordered on Oct. 12th and was delivered a couple days later.

Antenna. The last missing element of a robust program was an antenna. The initial antenna we considered was Winkler turnstile. We discovered that the vendor requires 28 working days to build the antenna and ship it. This was a major problem, given our projects schedule. Fortunately, we were able to find WiMo TA-1 antenna. While it is significantly more expensive (c.a. 100EUR) as compared to Winkler antenna, it has the great benefit of being readily available. The vendor claims the antenna will be shipped within 4 working days. The order has been placed on Oct. 22nd and we received a tracking number for the shipment. As of Nov 6, the package has arrived to Wroclaw and is expected this week.

The deliverable of this task was defined as to have all the hardware components received. The status of this task is **complete.** The engineer responsible for this task is **Tomasz Mrugalski**.

2.3 Early Dipole Antenna

The initial set-up was using a cheap (~4 EUR), low quality telescopic antenna that was included in the SDR package. This was a telescopic two arms in-door antenna that was originally intended to receive radio and TV broadcasts. The antenna was configured to work as a dipole. The telescopic arms were extended to 53.4cm length and set up 120 degrees apart and was positioned on the horizontal plane. This basic setup should in principle allow to receive signals with circular polarization. The antenna tripod with flexible arms has been used to hold the antenna in place, on top of a reasonably sturdy photographic tripod.

The antenna in its assembled state is presented in
Figure 2 below. This type of mounting posed several issues. First, the antenna itself was mounted to a tripod using small 10cm tripod with flexible arms made of foam/rubber-like material. It was wrapped around the head of a photographic mast that was part of much larger tripod. This junction was wobbly and there was no way to tighten it. It was stable during windless weather, but a bit stronger winds caused the joint to twist and bend and get loose over time, further decreasing stability.

The second problem was the large tripod. While it was used many times outdoors and was reasonably stable (including use for long time night astrophotography exposure), it was not well suited for long term outdoor usage, especially in poor weather. November is a period of poor weather in Gdańsk with strong winds, rain and occasional snow and blizzards. The tripod fell over several times. The fall could damage the antenna, but also damage the cables. After the tripod fell, the whole antenna with the tripod lying on its side was dangling on the cable and swinging intensely.

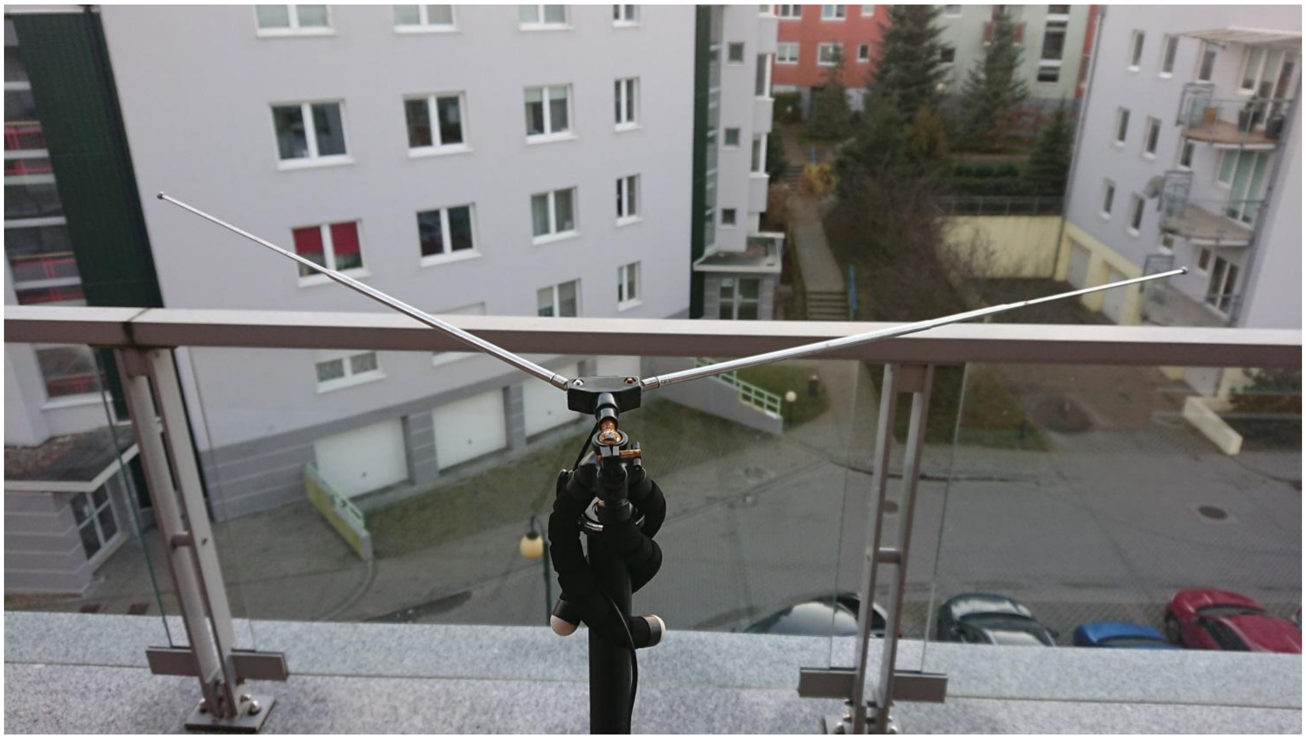


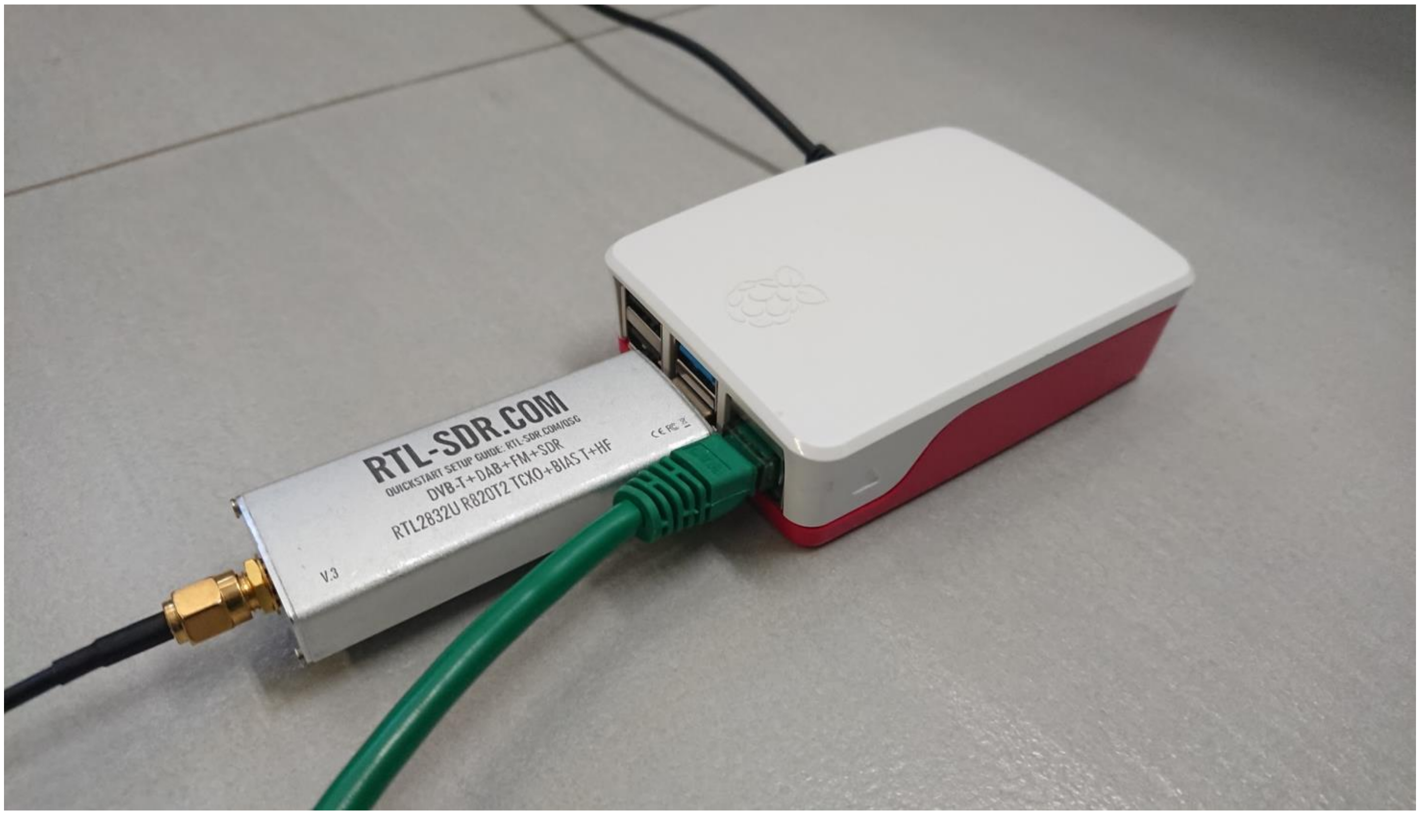
***Figure 2**:V dipole antenna*

We installed several software packages: *GNU Radio*, *GQRX* (both used to control SDR hardware), *gpredict* (a software that tracks satellites and informs about upcoming fly-overs), *NOAA-APT* (an open source alternative to *wxtoimg* software, it takes the recorded WAV audio file and attempts to extract image data from it).

2.4 System Integration

During the week of Oct. 21st our team did not do any individual tasks. Instead, we met together and spent half a day assembling the system. The old Raspberry 1B model has been replaced with the new Raspberry Pi 4B platform. The power system has been replaced as well. An old no-name USB power supply has been with a new USB-C dedicated power supply (branded as Raspberry Foundation recommended model). It was noted in many discussion forums that cheap power supplies, especially models used as phone chargers, do not behave well, especially under heavy load. In such cases the voltage could drop to a level that causes Pi to reboot. We have observed this issue once. Once migrated to the new Pi4 + new power supply setup, the problem did not occur any more.

We also put the Pi motherboard into a case. The hardware setup was assembled at Tomek’s apartment. One major problem to solve was how to deploy the system in a way that has good sky visibility from the antenna point of view, has Internet connectivity, has a power supply and the electronics is protected from the weather. After several attempts, we came up with a plan to house the system in the apartment close to a window. The SMA coax cable will go outside through not completely shut down window. The antenna will be deployed on a photographic tripod, standing on a balcony near the window. The assembled Pi system with connected SDR is presented in
Figure 3 below.


RTL-SDR.COM
QUICKSTART SETUP GUIDE: RTL-SDR.COM/QSG
DVB-T+DAB+FM+SDR
RTL2832U R820T2 TCXO+BIAS T+HF
V.3
Figure 3:*Assembled reception unit*

2.5 Initial Operational Capability

We experimented with several fly-overs. We later determined that the initial failures were caused by an attempt to receive transmissions from fly-overs that were low on the horizon. This, together with a poor antenna and a poor weather (it was foggy and rainy on the day of experiment), caused the signal to be too weak for our system. However, once we picked up a fly-over with high maximum elevation (almost crossed zenith), we were finally able to set up appropriate frequency for NOAA-18, record received transmission as audio and store it as a WAV file. The file was then processed using NOAA-APT software and generated image presented in *Figure 4* below.

The lower part of the image is garbled, because we went into NLOS (non line-of-sight) mode (part of the sky was obscured by the roof). Nevertheless, we consider this experiment a full success.

The deliverable of this activity is an integrated, functional ground station. The status of this task is **complete**. The engineer responsible for this task is **Ewelina Omernik.**

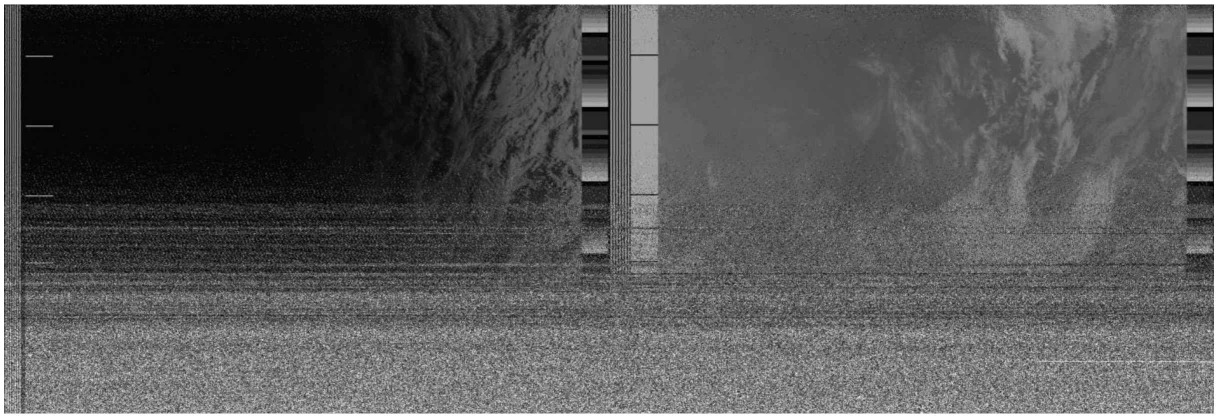


***Figure 4**:First received and decoded image*

2.6 Migration to Crossed Dipole Antenna

The new antenna had arrived in mid-November 2019. This is a crossed-dipole, circularly polarized antenna, specifically designed for satellite reception. Major parameters of the antenna are presented in *Table 2* below.

Parameter	Value
Vendor	WiMO
Model	TA-1
Frequency	135-152 [MHz]
Polarization	Circular right-handed
Impedance	50 [Ohm]
Standing Wave Ratio (SWR)	Less than 2.0
Dimensions	1300 x 1065 x 1065 [mm]
Weight	2 [kg]

***Table 2**: Major properties of the WiMO TA-1 antenna*

While much more powerful, the WiMO TA-1 antenna posed a number of challenges. First, it required assembly, but the assembly instructions were in German only. Second, the rods that were supposed to be fitted into holes that were too tight. Third, the antenna didn't work electrically. The details of how we solved the issues are discussed in Section 2.7 below.

Fully assembled antenna is presented in *Figure 5*.

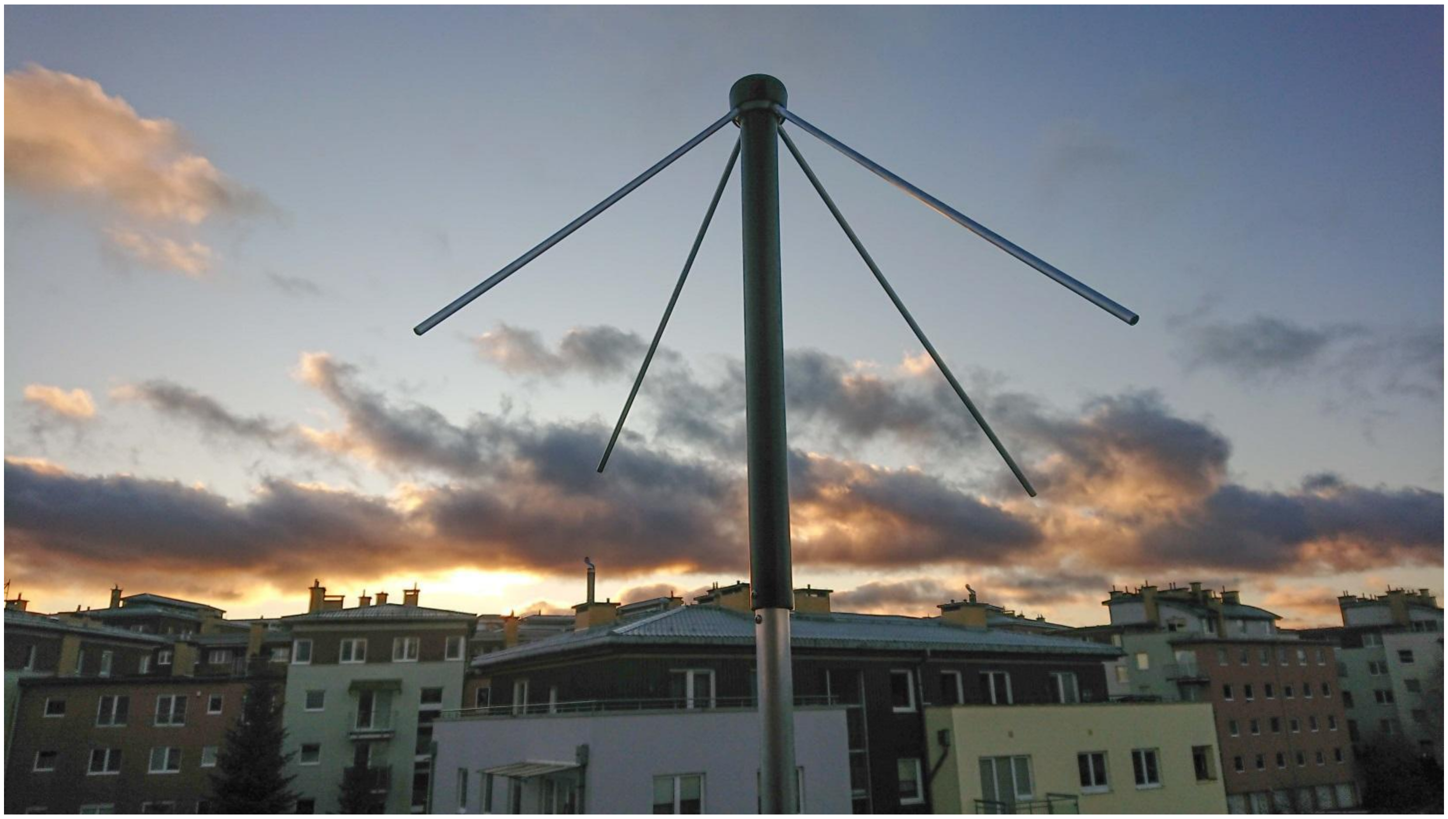


Figure 5*:Assembled crossed dipole antenna top*

2.7 Solving the Crossed Dipole Antenna Problems

The new antenna was shipped in a robust carton case that required some assembly. It came with an instruction that sadly was only partially translated from German to English. The assembly instruction was only a couple sentences that could be translated via Google Translate. It consisted of several basic steps, with only one possibly tricky part. The crossed dipoles (four rods to be assembled at the top) were of uneven length. Two longer and two shorter. These were to be installed alternatively. That way they formed one longer dipole and another shorter. The remaining steps required inserting two 106cm long reflector rods into crossed holes in the main vertical pipe. This step caused one of two significant issues we encountered.

The lower part of the antenna consists of a hollow mast made of an aluminum pipe with two horizontal 8mm reflector rods that are supposed to go through holes in the vertical mast. The problem was caused by the holes drilled by manufacturer were exactly 8mm in diameter as were the rods. The rods were very tight and it required significant force to push them through. Sadly, when a force is applied to a pipe that is not completely parallel to its axis, the pipe can bend. When a pipe is bent even slightly, it changes its cross section shape to no longer form a perfect circle, but an ellipsis. This shape never returns to its circular shape, even if bent back. This caused the reflector to get stuck in a very unfortunate position. That blocked the partially inserted rod to get stuck. It was not possible to either push it through or pull it out. After over an hour of futile attempts to remove it (that involved a vice, a hammer, a power drill, sandpaper and even a blow torch), a decision has been made to cut the last 3 cm off.

Once the reflector had been removed, a power drill has been used to widen the holes slightly. With that adjustment, it was now easy to place the reflector rods in correct position, The next minor problem was caused by the screws provided by the vendor. With the slight wider holes, they no longer were able to tighten completely. Fortunately, this was easy to solve with new screws of equal diameter, but a couple mm longer.

The third and last issue posed the biggest challenge. The antenna came in without any connectors, just had a dangling concentric wire. The N connector was assembled by a local CB radio service that had an experience with dealing with various antenna installations. Unfortunately, after connecting it to our setup it performed very poorly. We were able to receive popular FM radio stations operating in 89-105MHz band, such as Radio Zet, RMF or Anty-radio. However, we were able to receive only weak noise in the 137MHz band. That was puzzling, as the manufacturer specifies the working band of the antenna to be 137-152MHz, so the antenna should perform better in that range as compared to lower band. After many failed experiments, we asked prof. Zieniutycz for guidance. We were granted access to laboratory with vector network analyzer Anritsu MS2026B. After solving minor connection problems (both the antenna and the analyzer devices had female connectors), we were able to connect them and start investigating the problem.

One fundamental parameter that defines how well an antenna performs is SWR, Standing Wave Ratio. It’s a parameter that defines how much of the power provided is bounced back rather than transmitted, effectively measuring signal losses. For ideal antenna that transmits or receives 100% power the SWR would be 1.0. For a high quality finely tuned antennas, the parameter can be as low as 1.3 or 1.2 and increases upwards for worse antennas. While the specific requirements vary depending on the usage scenarios, in general antennas with SWR greater than 2.5 are deemed unsuitable for use. Sadly, our initial measurements showed the antenna had SWR of over 10. When calculating a reflection coefficient (Γ), we got the following value:

$$\Gamma = \frac{SWR - 1}{SWR + 1} = 0{,}8181$$

This means that over 81% was reflected back and on 19% of incoming signal was converted to electric signals. This indicated the antenna is broken and couldn’t be used in our set-up.

Fortunately, during the experiments in the lab the SWR diagram changed dramatically for a brief second, before reverting back to poor. This caused our team to look at what could cause the problem and after several attempts we figured out that the cable is broken several centimeters after the connector. It looked fine visually, but when twisted gently, there was very noticeable discontinuity. It seems the cable was damaged and the wiring was conducting only when the cable was bent one specific direction. The measurements in the good position are presented in *Figure 6* below.

Once the cause of the problem was discovered, the fix was simple. The cable was cut several centimeters above the faulty place, we installed the plug again and the antenna finally started working.

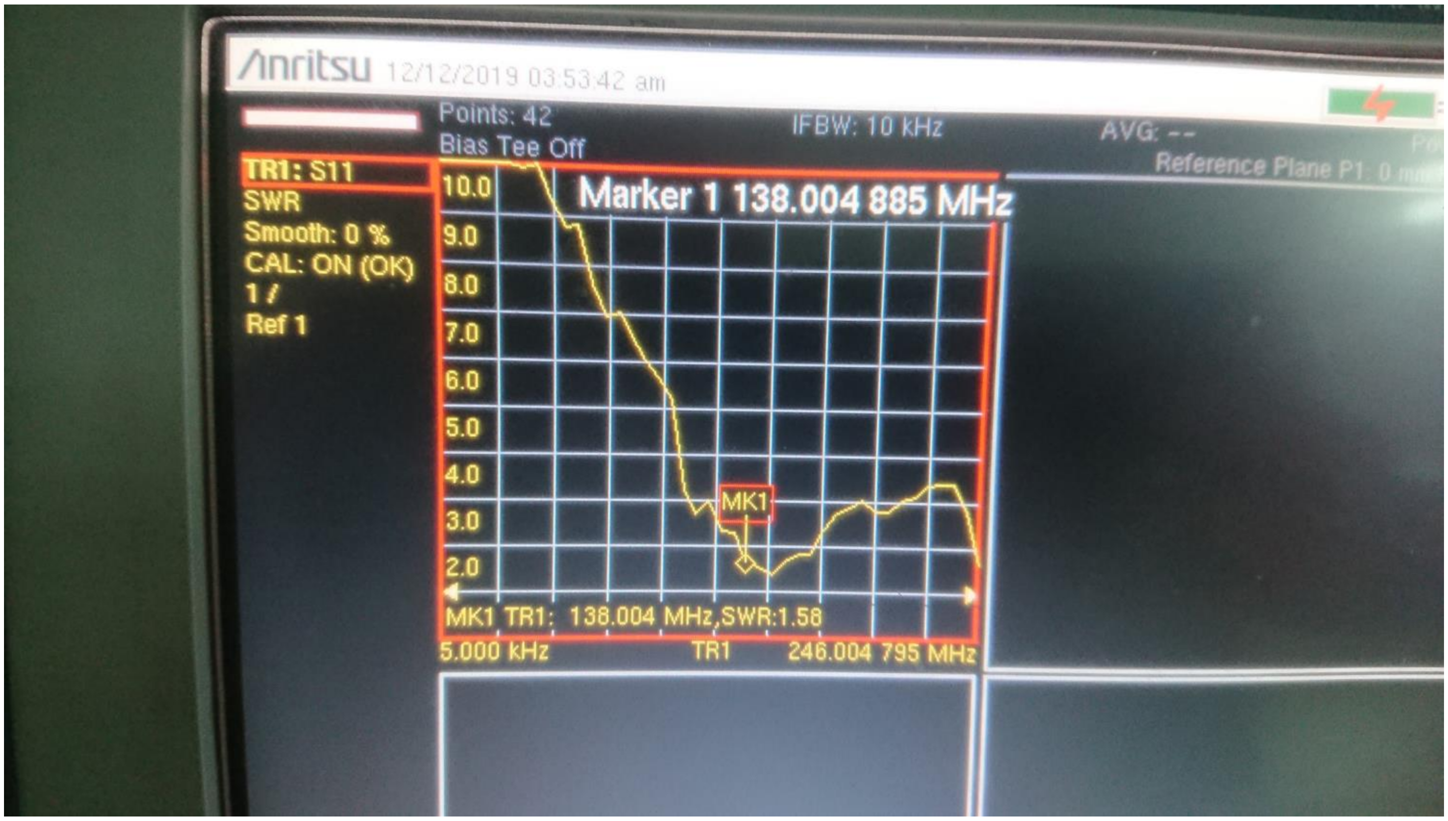


***Figure 6**:SWR measurements using Anritsu MS2026B vector network analyzer*

The last problem to solve was to mount the antenna in a way that is no longer susceptible to weather, especially the wind. Fortunately, the balcony has a very stiff and stable railing that could be used as a base. Using two metal clamps we managed to put the antenna firmly in place. The installed antenna base is presented in *Figure 7* below.

The antenna has been assembled and mounted on a balcony. Due to more permanent (or at least long term) nature of this antenna, a recommendation has been made by dr Siwicki to use an N-connector, a much more robust regarding resistance to adverse weather conditions as compared to SMA connector.

2.8 Software automation design

As of November 2019, the entire process of receiving satellite images is carried out manually. This requires project staff involvement at a specific time when the satellite is visible. It is burdensome and sometimes even impossible to conduct (e.g. when the fly-over happens during late night or early morning). Therefore an obvious decision has been made to automate the whole process.

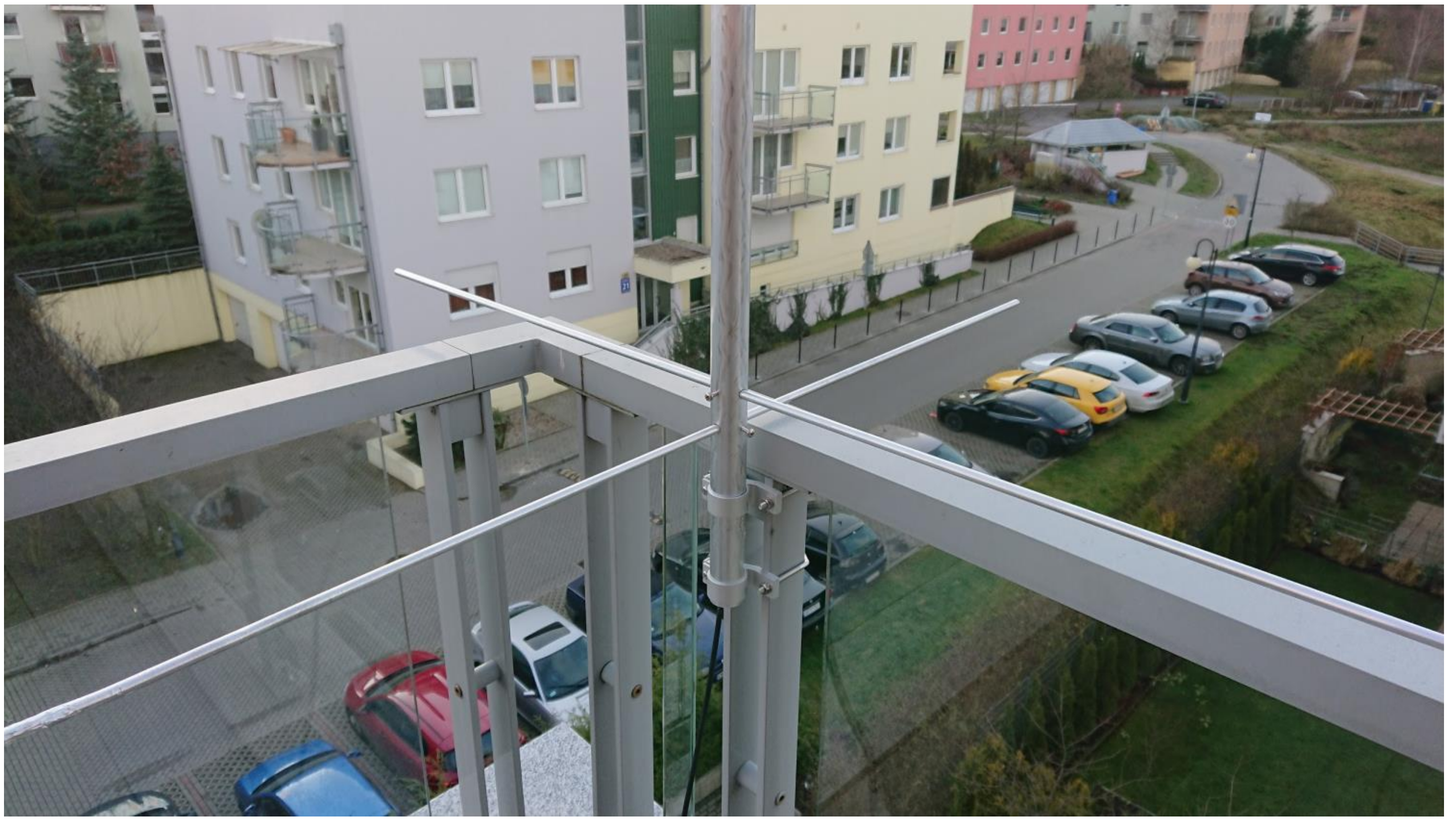


Figure 7:*Assembled antenna mount point*

The first step towards automation is to determine which tasks should be performed in which order to obtain the satellite images. As we have already done this manually before, we know how such a process should take place and what components are going to be required. A high level architecture of the software is presented in *Figure 8*.

Satellite tracking component. The satellites orbit the Earth, following a specific, predictable route. To capture a satellite signal, it must be in the visible range. Therefore we need a component that will track the movement of the chosen satellites and inform the system when the satellite will be in the field of visibility and when it will disappear. This component would be responsible for initiating the entire process and for stopping it.

Satellite data storage component. The tracking component reports the upcoming flight. However, information about the signal frequency, bandwidths, etc. is also required for data reception. These data are found on the Internet and are changing very slowly. The change is related to a slow orbital drift due to atmospheric drag, solar wind pressure and similar low impact factors. Each satellite has its own unique frequencies on which it broadcasts. Exceptions may occur when one communication module is damaged or a new one is added. Due to the variability caused by the above being low, we came to the conclusion that we can download the data just once and then store it. Assuming a long term evolution of the project, the system may be extended to do infrequent periodic updates of the orbital geometry. For the time being we assume this is out of scope.

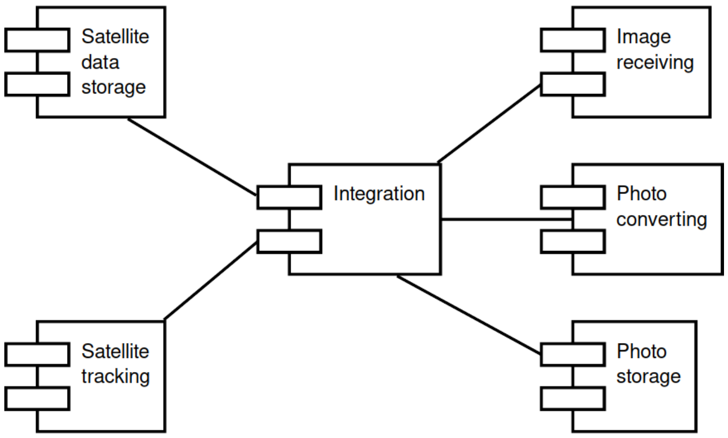


Figure 8*:High level software architecture*

Transmission reception component. The data provided by the components mentioned above will be used to control the actual transmission reception component. This component will use the SDR to tune in to specific frequency, decode the transmission using FM modulation and record it using a standard WAV audio file format.

Data processing component. The image data is encoded using Automatic Picture Transmission (APT) mode. It was used by several US and Russian satellites, but as of 2019 there are only three active NOAA satellites that use it. For more details, see [5] and [6]. There are several tools that offer APT decoding. For some time, the most popular was *wxtoimg*, however, due to the software being abandoned by its developer and unclear licensing status, we chose not to use it. noaa-apt, an open source alternative has been used instead [7]. The software is able to decode WAV files and extract image data from it. The output generated is a regular PNG file.

Photo repository component. Received photos need to be stored somewhere. We decided to designate a separate component that would provide the functionality of a repository for the received photos. One specific aspect of consideration here is that the platform chosen (Raspberry Pi) has limited disk space available. The SD card used is 16GB, but part of it is already taken by the operating system and installed software, so the actual capacity available for the data storage is smaller. Since there is a powerful NAS storage available in the LAN this ground station is connected to, a network mounted storage is likely to be used.

Integration component. We also need a component that connects everything together. It would be responsible for communication between all of the other components.

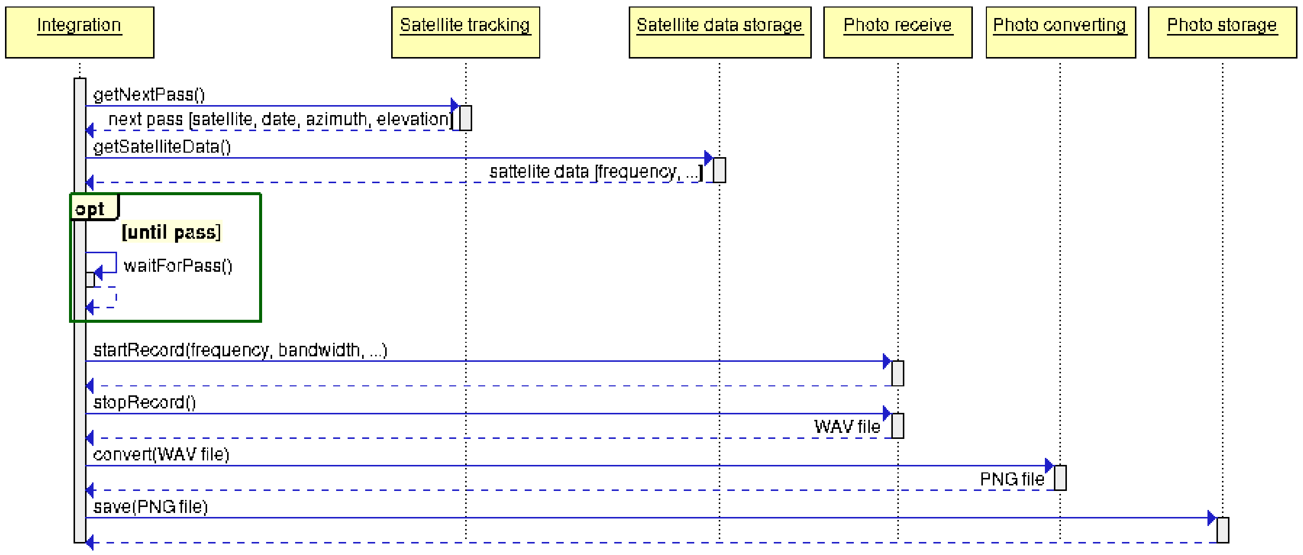


Figure 9*:Control flow between components*

An interaction between different components will be conducted sequentially. The control flow will start and end with Integration, which will call specific routines from other components. The control flow is presented in *Figure 9* above.

A separate document “Image reception automation” has been dedicated to this task. See appendix A3 for details.

The deliverable for this task is to have a high level design for the automation solution. The task is **complete**. The engineer responsible for this task is **Sławomir Figiel**.

2.9 Software implementation

We started planning our software work by doing a research of available existing solutions in the Internet. We found a few ready solutions to receive APT transmissions used by NOAA satellites. We analyzed technological stack of them. It allowed us to discover some interesting components. However, we noticed that all existing complex solutions use *wxImage*. We decided to abandon this program, because it is no longer available, as the authors changed the license to closed source. Other problem was specialization of these solution only in NOAA satellites.

We chose Python as the main programming language. Its main advantage is fast development, easy integration with other programs, and great base of ready components.

In testing phase we used *gPredict* to track satellites. But it is GUI application and it isn’t dedicated to automating tasks. Existing solutions used *predict* program. It is a command line application and provide server to share transition data. During test of *predict* we noticed some problems. The output of the server was human friendly, but difficult to parse by a computer. We needed custom processing for each used command. The main disadvantages were problems with generate configuration files. They were not created or had invalid content in our environment. Therefore we decided to not use it. We used *orbit_predictor* Python package [11]. It has a simple interface and quite accurate prediction algorithm (~5 second difference from gPredict).

We didn't use a specialized component to storage satellite data (as transmitter frequencies). In prototype we support only NOAA satellites, therefore we put all needed data in Python dictionary.

In initial version our photo repository was SD card of Raspberry PI. Later we wrote a script to upload data to the content share server. The upload script is described in Section 2.12.

The transmission reception component took most time to automate. In test phase we used GQRX. It has a rich graphical user interface. An example screenshot is presented in *Figure 10* below. We have real time preview of transmission state and we may manually adjust receiver. But in automation flow program must set correct parameters without human interaction. We needed to determine optimal values (sample rate, gain and others). For control SDR we used low level program: *rtl_fm*. It set receiver to specified frequency and simply perform FM demodulation. Next we used *sox* program to process raw signal, extract single channel and save in WAV format.

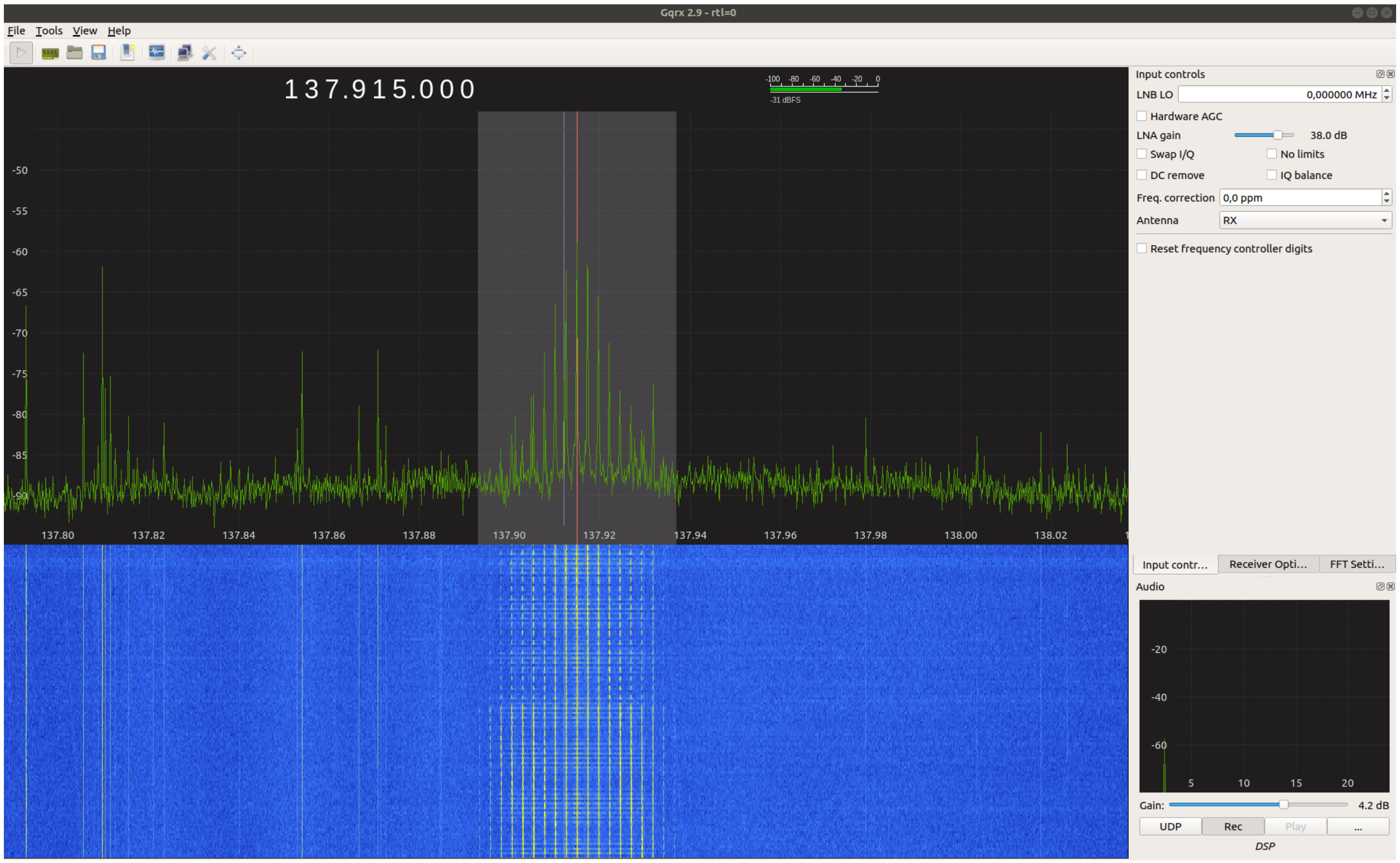


Figure 10:*A screenshot of GQRX software during strong transmission reception*

WAV file was passed to *noaa-apt* software. It is a modern alternative for discontinued *wxImage*. It is designed for radio amateurs. It contains a custom NOAA decoder which is less susceptible to noise. It also has a command line interface. As output we got a PNG file with decoded image.

NOAA imagery was then passed to separate script responsible for delivering data to content share server. The *submit_obs* script is described in Section 2.12 below.

The main task of our script was scheduling reception during upcoming passes. It used to track satellite component to get the list of upcoming passes. Based on transition parameters (azimuth, elevation, duration) it chose one pass and plan processing flow at specified time.

The script is fully operational. However, it needs a trigger to start. It may be run by user or CRON (scheduling tool for Linux). But these solutions aren't ideal and require care by administrator. We plan to use SystemD capabilities and write compatible service. Then software will be more autonomous, start with operating system and reset if any problems are encountered.

The task is **in prototype phase**. The responsible engineer is **Sławek Figiel**.

2.10 Test campaign

The responsible engineer is Ewelina Omernik. TODO. See gitlab #17.

TODO: Ewelina to put her tests description here.

2.11 Data utilization

A major part of the utility of this project is to make the received data available. There are several aspects that should be taken into consideration. First and foremost, the data should be made available to a wide audience. Second, the data should be accessible in a way that does not require any specialized hardware or software. Third, the data have a short period of usability, after which it becomes outdated, so it should be made available as soon as it is acquired. Fourth, the time our team could dedicate is limited by the number of other projects planned for this semester (8 projects in total). As such, the solution should be reasonably easy to do in terms of engineering effort required to complete it. To address all of those requirements, several possible publication venues were considered. One way would be to upload the produced data to a file storage somewhere. This would address the need to make it available promptly, but would not make it easy to consume by end users. It would be somewhat difficult to find as well. The second alternative considered was a written report. This would probably be the easiest way to make the data public, but it wouldn't achieve much. The data would be instantly outdated and nobody would realistically use it. As such, that alternative was rejected as well. Finally, the last considered approach was to develop a web service that would be able to host observations and make them available as soon as they're decoded. This approach also addresses the requirement of being easy to use (only a web browser needed), and being available to a wide audience. Since this approach has most benefits compared to alternatives, it was chosen as the most appropriate solution to the problem at hand.

2.12 Data Processing Pipeline

The following section describes data flow in the system. The overview of the system has been presented in *Figure 11* below. The yellow arrow represents electric signal received by the antenna that is sent over coax wire to the hardware SDR. The red arrow represents control commands and the 1MHz bandwidth that is processed by the *gqrx* or *rtl_fm* software. The blue arrow represents a WAV data being generated by the *gqrx/rtl_fm* software and decoded by the *noaa-apt*. The white arrows represent various types of data, including the decoded PNG image, system logs, timestamps and other information available. The green arrow represents a secure SSH connection.

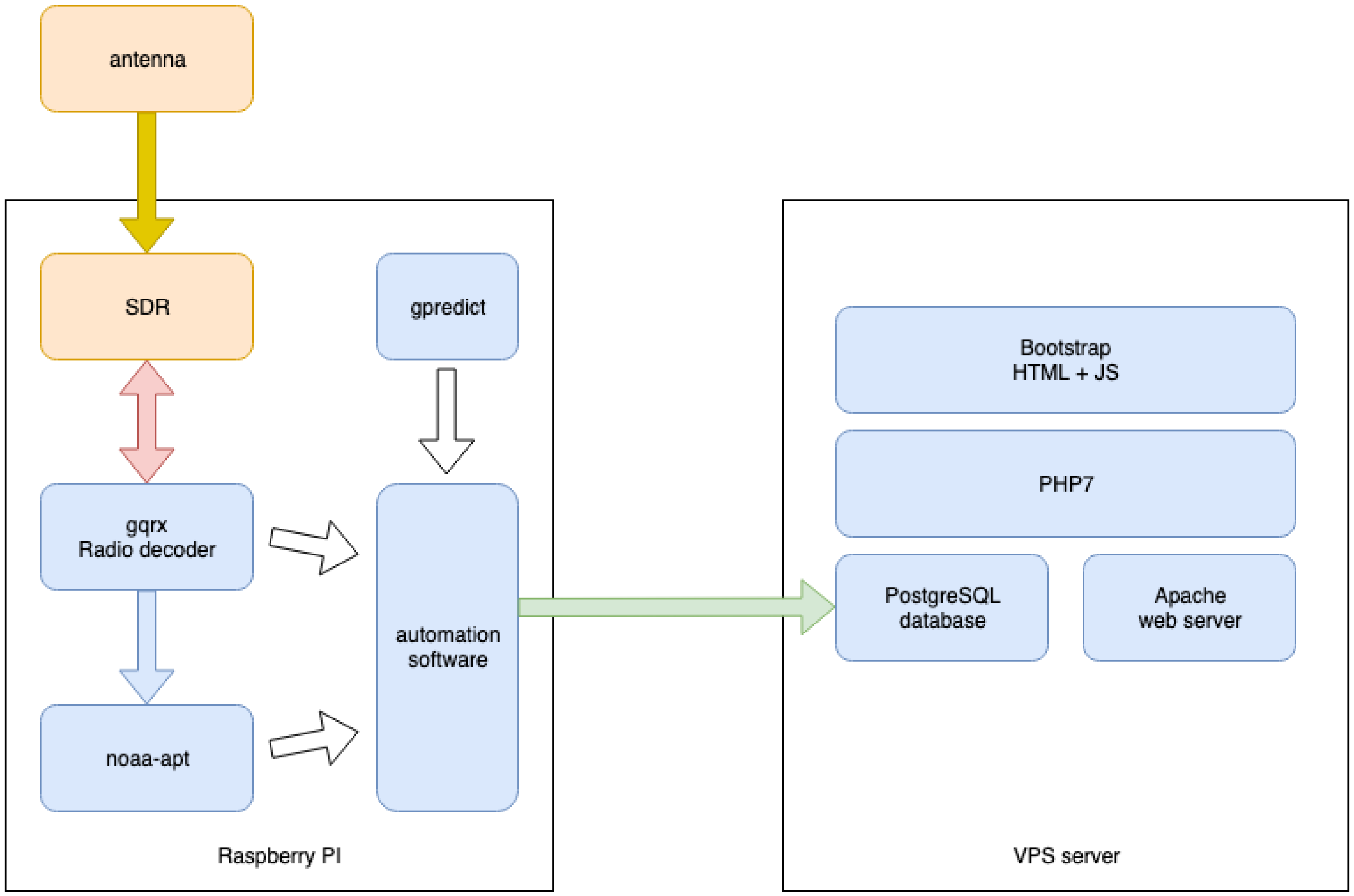


Figure 11:*Overall architecture of the data processing pipeline*

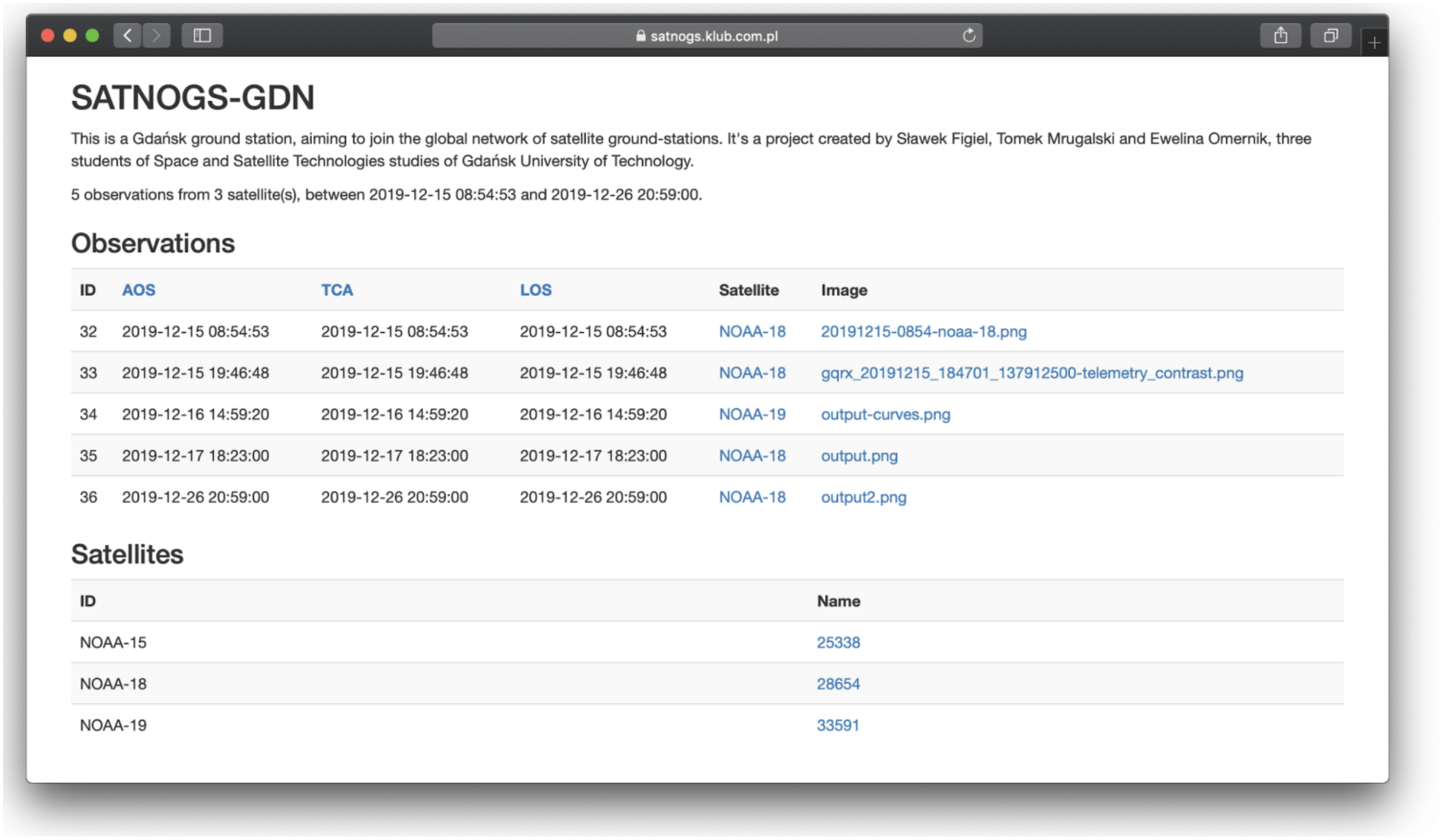
The ground station is located at Tomek’s apartment, which has reasonably good residential connectivity with good downlink capability (reliable 150Mbps), but rather poor uplink that varies a lot. This is insufficient to host a reliable public service. As such, a decision has been made to set up a server on one of the hosting services. OVH has been chosen as a provider due to their affordable prices, IPv6 connectivity and ease of setup. The server is running on Debian 10 GNU/Linux system. The type of service chosen is VPS (a Virtual Private Server).

Once the resulting image is generated, the data needs to be exported to the web service. A Python script called *submit-obs* has been developed that takes care of this process. It reads the local PNG file, the transit times (AOS, TCA and LOS), any additional log file and sends them to a remote web server over secure SSH channel. The confidentiality is provided using ChaCha20 [9], with authentication using 128 bits elliptic curve 25519 keys [10].

A PostgreSQL 11 database has been set up to handle the observation data. While the schema is rather simple at this stage of the process, the system is expected to get much more complex in the future, thus warranting a full relational database.

The web service is handled by Apache 2.4, a popular web server with PHP 7 extensions. The website itself is written in PHP 7 with Bootstrap library being used for end-user visual elements.

The web interface is available at https://satnogs.klub.com.pl. An example view of the page is presented in *Figure 12* below.



SATNOGS-GDN

This is a Gdańsk ground station, aiming to join the global network of satellite ground-stations. It's a project created by Sławek Figiel, Tomek Mrugalski and Ewelina Omernik, three students of Space and Satellite Technologies studies of Gdańsk University of Technology.

5 observations from 3 satellite(s), between 2019-12-15 08:54:53 and 2019-12-26 20:59:00.

Observations

ID	AOS	TCA	LOS	Satellite	Image
32	2019-12-15 08:54:53	2019-12-15 08:54:53	2019-12-15 08:54:53	NOAA-18	20191215-0854-noaa-18.png
33	2019-12-15 19:46:48	2019-12-15 19:46:48	2019-12-15 19:46:48	NOAA-18	gqrx_20191215_184701_137912500-telemetry_contrast.png
34	2019-12-16 14:59:20	2019-12-16 14:59:20	2019-12-16 14:59:20	NOAA-19	output-curves.png
35	2019-12-17 18:23:00	2019-12-17 18:23:00	2019-12-17 18:23:00	NOAA-18	output.png
36	2019-12-26 20:59:00	2019-12-26 20:59:00	2019-12-26 20:59:00	NOAA-18	output2.png

Satellites

ID	Name
NOAA-15	25338
NOAA-18	28654
NOAA-19	33591

Figure 12:*Example screenshot of the web interface.*

The engineer responsible for this task is **Tomek Mrugalski**. The task is now **complete**, although future improvements to the presentation layer are expected.

3 Project summary

This section describes the project summary, conclusions and next steps being considered.

3.1 Conclusions

The acquired hardware and the developed software allows us to receive satellite transmissions in a reliable, repeatable and predictable way.

The quality of the received images vary greatly between fly-overs. The major factor determining the quality is maximum altitude. If the satellite is high over the horizon (45 degrees or more), the signal is strong and can be received long enough to produce good images. An example of such decoded signal is presented in *Figure 13*.

The team being a group of mostly software engineers, initially lacked experience with the satellite communication and antenna issues. Stepping outside of our comfort zone was a bit challenging, but ultimately very rewarding experience. We learned a lot and by proposing this topic for our group project and we strongly feel that we expanded our expertise. The project gave us a much deeper insight into the complexity of radio communication in space. Being able to interact with objects in space is something unique to our field of study.
We greatly underestimated how many aspects had to be taken into consideration. The antenna we purchased didn't work initially. Debugging the problem and finding the root cause of the problem was stressful, but a very good learning opportunity.

The biggest challenge faced during the project was not technical. The classes planned for this semester were very tightly packed. Furthermore, all members of the team are employed, so we all had to balance our responsibilities.

Finally, not every member of the team contributed to the project equal effort. In the management circles one of the basic rules is that people can be motivated with rewards ("a carrot") or with punishments ("a stick"). Running a student project had a very limited rewards available and no way to really discipline. This resulted in a tricky dilemma - more motivated team members could either take over the tasks that were running behind or let them slide. Taking over would lead to a fallacy of hard working members being punished with even more work and lazy members would be rewarded with fewer and easier tasks to complete. On the other hand, not taking over would risk the whole project success or at the very least degrade the general quality of the resulting product.

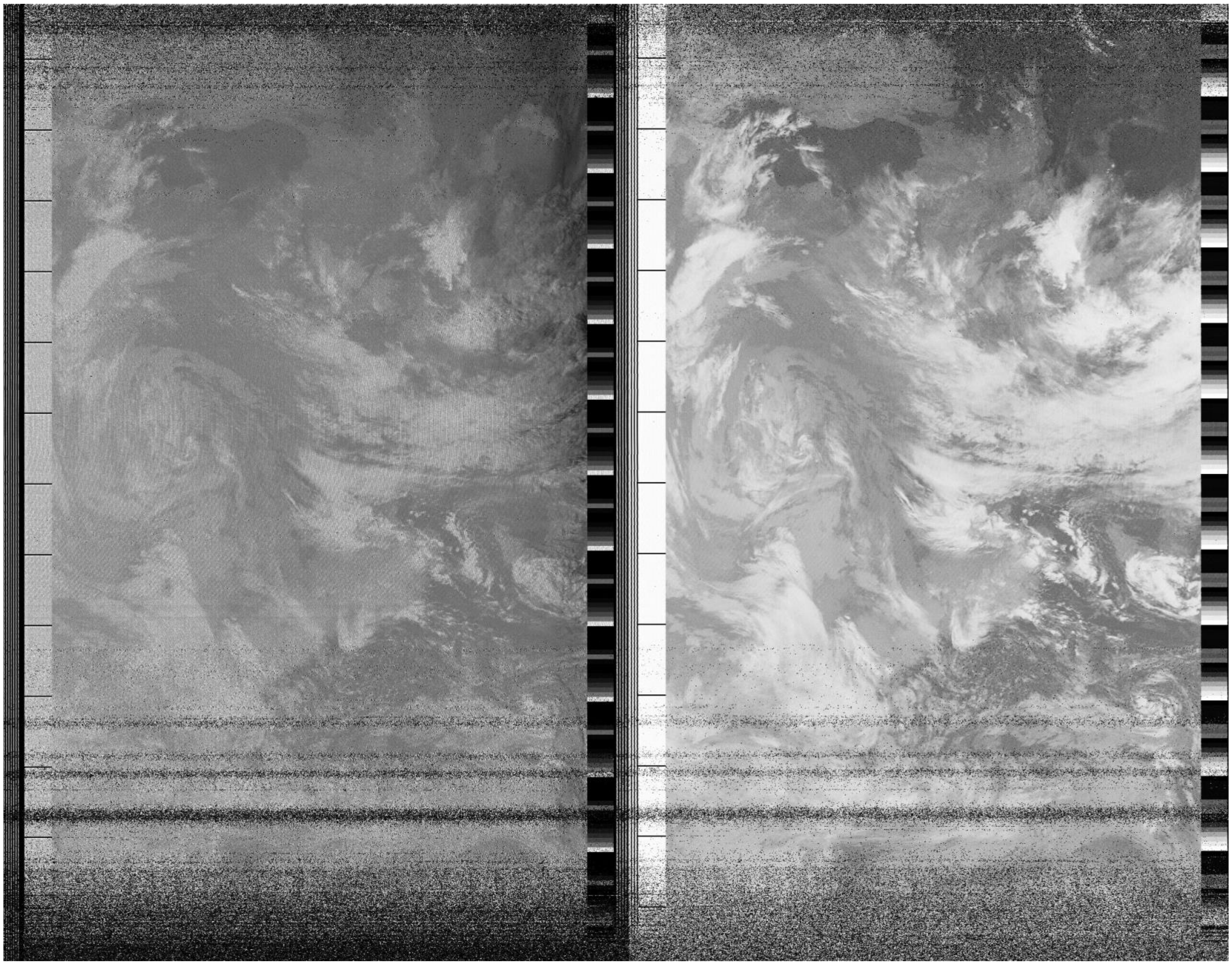


***Figure 13**: An example image received*

3.2 Next steps

Two members of the project - Sławek Figiel and Tomek Mrugalski - want to continue the project, beyond the submission deadlines imposed by the University. We have discussed a number of future improvements and extensions:

- develop a ***better web interface*** to provide more dynamic data. In particular, a simple list of observations will not scale well once the number of observations grows. For that purpose, we'd like to reimplement the website using some tools that provide data source handling with searches, ordering and pagination, such as Angular Material or PrimeNG. A good tool should also offer more complex data visualization, such as charts that could provide frequency of observations, convenient date selection and making other search criteria easily available.
- **Integrate with the SATNOGS network**. Right now our ground station is a solitary station operating on its own. There is a large and fast growing network of amateur ground stations called SATNOGS [1]. Integration with that network will give a much better exposure for our results. Also, it will allow correlation of our observations with other ground station, potentially even combining the data together.

- **Switch to directional antenna**. The antenna currently used is omnidirectional. That means it is static and is able to receive transmissions from all directions. A moving directional antenna has much better gain and is able to receive much weaker signals and thus can use higher bit-rate.
- **Investigate discrepancies between *gpredict* and *orbit_predictor***. It is puzzling why there are discrepancies in predicted fly-over times in the range of around 5 seconds. Both are supposed to faithfully implement SGP4 algorithm, published by NORAD [12]. Despite the SGP4/SDP4 models being published 40 years ago (1980), they are still considered the standard in the industry. All software faithfully implementing them are supposed to produce exactly the same data.
- **Switch to UHF and S band**. Right now our system is able to receive VHF, which is considered the entry band in the radio amateurs world. With higher frequency UHF, S band and beyond, the number of potential satellites to be received grows significantly. In particular, a Polish company called Sat Revolution, offers reasonably inexpensive UHF and S band communication modules for their cubesat platform. This lets us speculate that future Polish satellites are likely to use that specific band. As such, we should be ready to provide downlink capability.
- **Cooperation with amateur Cubesat projects.** The nanosats are becoming more and more popular. Many universities, schools and science organizations (as ESA) support students in the construction of own cubesats. However, building a satellite is only one problem. Another issue is to maintain connection with the spacecraft and receive data from orbit. Maintaining the network of ground station is such a demanding task as design satellite. We may take advantage of the unique Tricity location (almost maximal latitude and center longitude of Poland area) and support future student projects by offering downlink services.

3.3 Bibliography

1. Satnogs project website, https://satnogs.org, retrieved on 2019-10-14
2. Perun Rockets website, http://perunrockets.net/posluchajmy-satelitow.html, retrieved on 2019-10-14
3. ***Pobieranie zdjęć ziemi z satelity za pomocą anteny DIY i... tunera TV***, https://majsterkowo.pl/pobieranie-zdjec-ziemi-z-satelity-za-pomoca-anteny-diy-i-tunera-tv/, retrieved on 2019-10-30.
4. ***Simple NOAA/Meteor Weather Satellite Antenna: A 137 MHz V-Dipole***, https://www.rtl-sdr.com/simple-noaameteor-weather-satellite-antenna-137-mhz-v-dipole/, retrieved on 2019-12-11
5. ***Automatic Picture Transmission(APT)***, https://www.sigidwiki.com/wiki/Automatic_Picture_Transmission_(APT), Signal ID wiki, retrieved on 2019-12-11
6. wikipedia, ***Automatic Picture Transition***, https://en.wikipedia.org/wiki/Automatic_picture_transmission, wikipedia, retrieved on 2019-12-11
7. ***Noaa-apt***, project website, https://noaa-apt.mbernardi.com.ar/, retrieved on 2019-12-11
8. T.Mrugalski, S.Figiel, E.Omernik, ***SATNOGS-GDN project homepage***, https://gitlab.klub.com.pl:30000/astro/satnog-gdn, retrieved on 2019-12-11
9. Y. Nir, A. Langley: ***RFC 7539: ChaCha20 and Poly1305 for IETF Protocols***, IETF, May 2015
10. D. Bernstein, ***A state-of-the-art Diffie-Hellman function***, https://cr.yp.to/ecdh.html, retrieved on 2020-01-03
11. Satellilogic, ***Orbit_predictor***, Python package, https://github.com/satellogic/orbit-predictor, retrieved on 2010-01-07
12. T.Kelso et al., ***SPACETRACK REPORT NO. 3:Models for Propagation of NORAD Element Sets***, US Department of Defense, 1980

3.4 Appendices

A1. Weekly report – 2019-10-24
A2. Weekly report – 2019-10-31
A3. Image reception automation (a feasibility study)
A4. Crossed Dipole Antenna Evaluation (preliminary report)