2015 Cornell Cup Final Report

STRATUS

University of Colorado of Denver
Skyler Saleh
Antonio Duarte Jr.
Bruno C. Mary
Xin Li

Faculty Advisor: Dr. Dan Connors

Table Of Contents

Challenge Definition

The purpose of Project Stratus is to support the rapid construction of customizable robotic solutions. The Stratus system integrates innovations in robotic vision, navigation, and control systems to impact the development and deployment of robotics in society. The result of the design is a coherent framework that coordinates an array of collaborating system elements that collectively sense the surrounding environment and control physical movement. The project has the potential to impact developers seeking to create robotics solutions that benefit society.

Robotic Integration in Society

The list of potential aerial and ground robotic systems in society is vast and growing. Some current areas of development include agriculture, transportation, product delivery, social and medical assistance, and service robots for personal and domestic use. The next stage of robotics in society will involve more common and commercial aspects including journalism, photography, real estate, delivery, and local governments, agriculture and others embrace drone technology. At the same time, there are many barriers to the use of robotics that range from Federal Aviation Administration (FAA) regulations that restrict flight to commercial issues related to the insurance concerning autonomous cars. The foremost obstacle to large scale use of robotics in society is safety. While the first era of robotics was focused on developing inexpensive functional platforms, the next will focus on deploying easily controllable and autonomous solutions for a large number of unique problem domains.

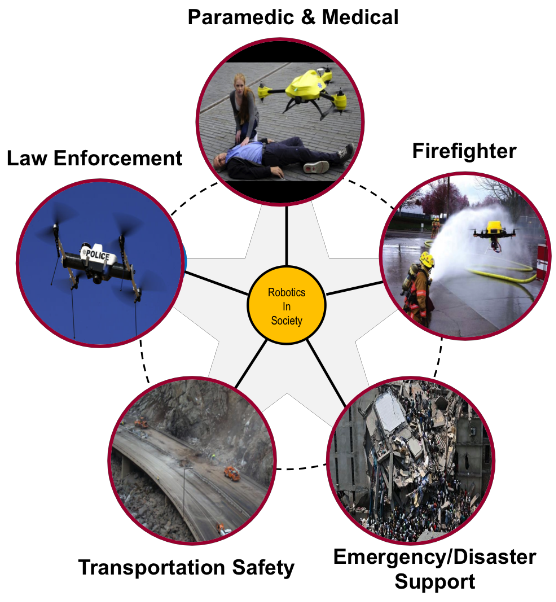


Figure 1: Integration of Robotics in Various Roles in Society

While there are interesting visions for robotic applications in the commercial domain, Project Stratus is founded on the goal of empowering robot designers that explore applications in critically important domain of society.   The purpose is to have society benefit the most by deploying robotic systems in areas that put men and women in potential danger, such as emergency response and safety scenarios. Figure 1 illustrates a broad spectrum of robotic roles in society in which people currently risk their lives helping others. Overall, the development of robotic solutions in these domains is hindered as design teams often must start from scratch when trying to meet the special requirements of their scenarios. For example, in emergency response cases, some common design goals are to provide fast-response and assistive services while for transportation safety systems, the goals are aligned for preventative actions predicted and determined from surveying. With the wide availability of rapid prototyping technologies coupled and the increasingly powerful yet inexpensive microprocessors, the general robotics field is at the frontier of a revolution of change. While such technologies play a critical role in building a robotic platform, the development time of a group to design, to test, and to deploy a solution platform in a specific domain area remains a fundamental problem.

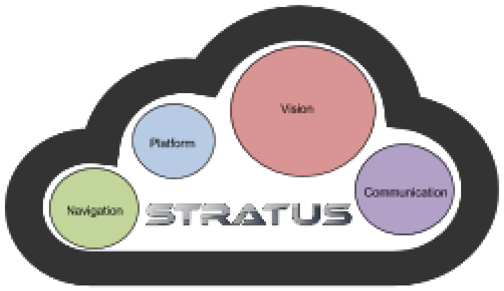


Figure 2: Stratus Solution of Vision, Navigation, Communication, and Platform

Proposed Solution
As there are distinct requirements for each specialized robotic application, rather than have each domain re-develop and construct an operating robotic core, the grand vision of modern robotics is to provide solutions for every domain. The Stratus solution provides an all-in-one feature-rich programmable robotic platform to meet the grand challenge. The Stratus framework is a collection of the fundamental components and computational elements for sensing, communicating, navigating, and controlling robotic vehicle platforms. In this way, the Stratus functionalities are strategically distributed and optimized between the robot platform and command station the eliminate the development burden and time between design and deployment. Figure 2 shows an overview of the core design elements integrated within Stratus: Vision, Navigation, Communication, and Platform. The modular design substantially lowers the development time by providing a flexible framework of open source and commercially-off-the-shelf (COTS) components including the Intel Edison and Intel HD Graphics.   To explore and test the versatility of Stratus, the platform is applied and demonstrated on both an aerial Hexacopter and a ground Mini-Rover. The measurable results of the Stratus project are based on its cost, volume, capabilities, programmability for domain independence, and application to a range of meaningful use cases. The result is a generation of more sophisticated robotics capable of sensing and navigating increasingly difficult environments.

Project Solution
Project Stratus is an integrated hardware and software framework for robot vision (sensing), navigation, and control. To support these three core functionalities, the solution is divided into four

interconnected and overlapping concepts: vision, communication, control, and platform. The project development includes the distributed communication linking the remote robotic platform and a mission control station. The project development is not restricted to one active robotic platform, however in practice, the design system was tested assuming an individual aerial or ground robot. The dual robot design scenarios allowed Stratus framework to address the general issues of building common control interface for operators of robotics.

Figure 3 shows a traditional robotic platform consisting of multiple sensors, communication systems, computational, and control systems. Typically an operator manages the system using a Radio Control (RC) transmitter that is commonly used by hobbiest but often creates a barrier. The RC transmitter communicates to the RC receiver on the robot and the commands must be processed by on-board computer that in turn relays commands to micro-controllers that convey information to the motor controllers. Concurrently, the robotic platform must sense using visual light and infrared (Kinect) cameras and potentially convey that data back to a wireless receiver. Overall, the design process involves multiple technologies coming from different suppliers.

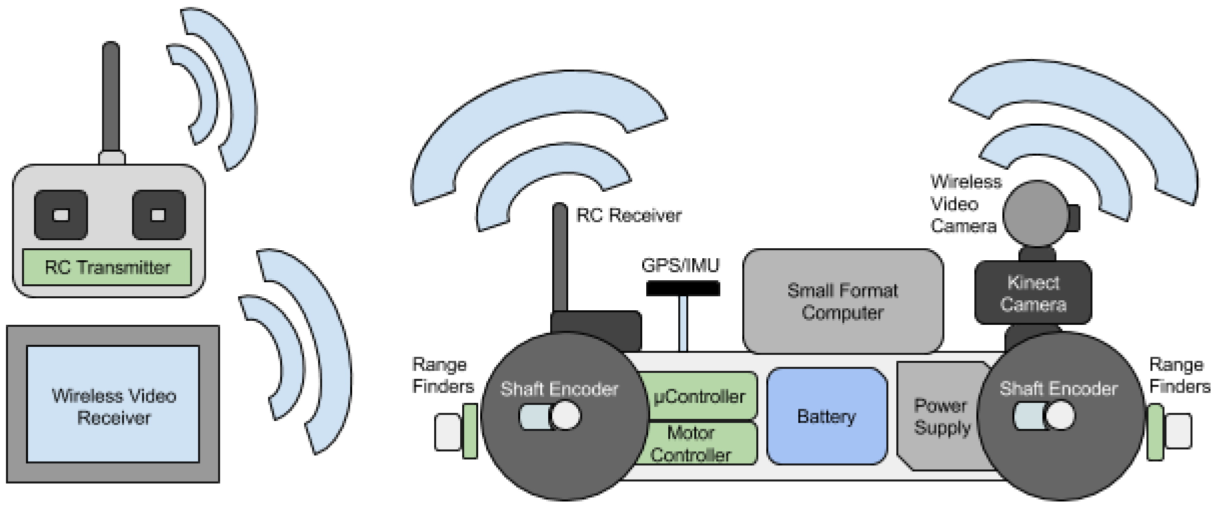


Figure 3: Traditional Illustration of Complex Robotic Platform

Overall, based on the traditional robotic platform of Figure 3, multiple improvements were envisioned:

- Usable control system- provide interface without require extensive experience by the operator.
- Communication- construct a seamless and efficient framework for transmitting and receiving between the robot and the command station.

At the center of these concepts is the need for a framework that couples innovation with the strategy of integrating numerous components.

Integration

The integrated approach of the Stratus solution simultaneously addresses the challenges of specialized robotic design imposed by a number of problem domains with varying characteristics. Figure 4 shows the framework deployed as a new modular system. The focus to integrate the multiple components into a modular form allows a broad set of designs to leverage the system. Overall, the integration improves the design space of robotics by eliminating the development time for

In this way, a small game controller uses BlueTooth communication to connect to a local laptop computer that is provides the capability of transmitting and receiving all data streams (control, navigation, sensing) with the remote robot. The on-board Stratus module provides the all-in-one communication unit and the coordination of multiple sensors. The vision in mind was to have the Stratus platform scale across the number of desired sensor inputs. The design is tested with streaming of both visual light and infrared (depth sensing). The overall result of the robot with Stratus is a streamlined system that has been optimized to provide low-overhead response with the operator as well as integration with automated navigation controls such as Global Positioning Systems (GPS).

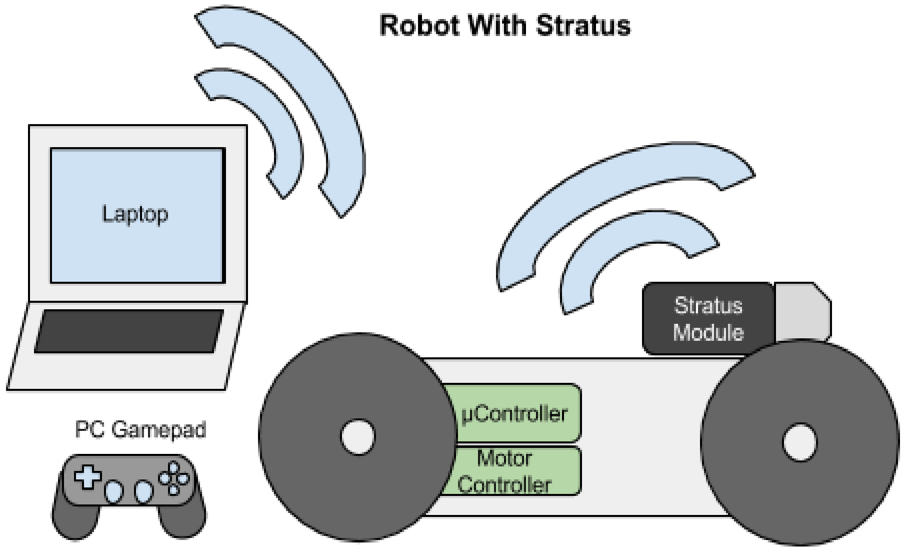


Figure 4: Stratus Project Simplification of Modular Robotic Components and Control

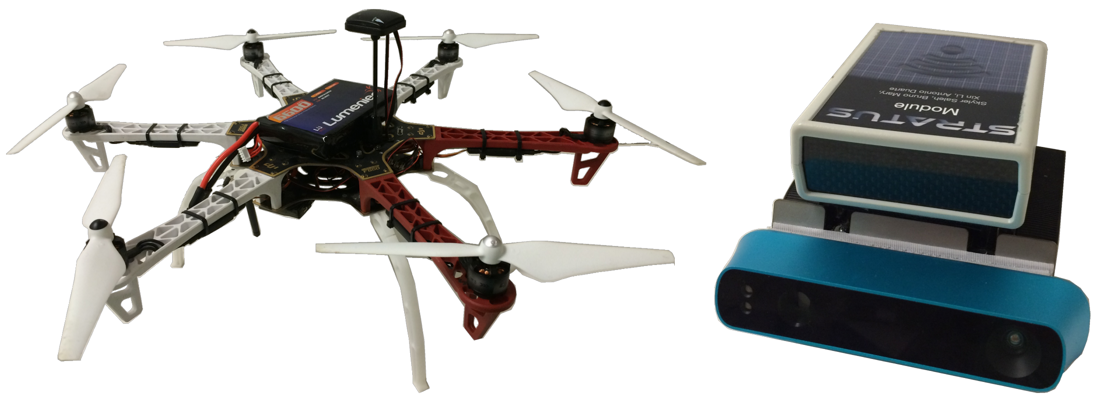


Figure 5: Stratus Module Deployment in Hexacopter and Mini-Rover Platforms

Customer Value Proposition

The Stratus project originated with the goal to develop a custom solution for the integration of a mapping and navigation module to supplement actual ground and flying robotic technology. The project initially was based on a single use of a custom developed Simultaneous Localization And Mapping (SLAM) algorithm for navigation and scanning but was broken into two single processes (navigation and mapping) to fit each requirement. The navigation process uses a simplified form of the slam which provides higher rate of data to the control module while the mapping algorithm is computed in a ground station to create a three dimensional display of the scan.

The Stratus developed module offers an all in one solution for ground and aerial platforms users that allows a simple and cost efficient interface to increases the self sufficiency of autonomous robots in crowded or open environment. Figure 5 shows the Stratus modules deployed in dual environments of a Hexacopter and Mini-Rover. The Stratus module raises the bar in navigation solution by adapting to changing environment such as changes that occurs in enclosed area such as people moving in a room or moving furniture. Outdoor most navigations depends of global positioning system and satellite mapping while the Stratus module relies on visual actual data to navigate and scan.

Scope Changes

The original design of Stratus was more narrowly designed only for integrating a depth-sensing vision system into an autonomous aerial vehicle. One key difference with the original design proposed relates to the amount of on-board computational processing available to robotic platforms.

Computational Requirements of SLAM: Stratus Distributed Computing Evolution

Overall, the field of computer vision seeks to develop in machines the capabilities that currently exists only in animals: visual intelligence. Important aspects of human vision such as the ability to identify obstacles, plan paths around them, learn from one's surroundings, in real time, are abilities crucial to our ability to navigate and interact with the real world. Yet, replicating similar behaviors in robots, a process called Simultaneous Location And Mapping(SLAM), has proven to be notoriously difficult. While, there is many different techniques for doing SLAM, no current implementation is well suited for most robotic applications.

However with the recent advances such as affordable 3D depth cameras, powerful GPU's, and new SLAM techniques, it should be possible to build a SLAM system that is well suited for robotic applications. Thus the original goal of the project was to generate SLAM results on the robotic platform. Figure 6 illustrated the initial design carrying an NVIDIA Tegra TK1 processing system with programmable Graphics Processing Unit (GPU) support. The computational requirements of SLAM are summarized as:

- 25 Update frames per second
- ~40ms of Latency
- < 1cm/meter positional drift
- < 5°/revolution rotational drift
- 4 bytes 1cm^2 of surface area

The summary of the initial investigation of SLAM on a mobile platform found that the two major components would require for Pose estimation of 200 GFLOPS (Billion floating-point operations per second) and Database fusion of 40 GFLOPS.

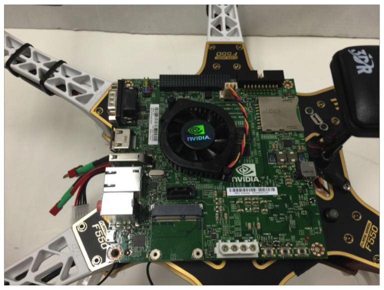


Figure 6: Original Hexacopter with NVIDIA GPU-enabled Design

Similarly, the next prototype design to explore capabilities of the Stratus concept used a iPhone development as the communication and processing system. Figure 7 shows the prototyping process with team member Bruno Mary customizing the design using a combination of SolidWorks CAD and plastic machining. The team used 3D printing technology of a MakerBot Replicator 2X to enable highly customized cases. The design cycle in exploring these prototypes and constraints, while giving the team experience, required approximately 4 months of time. The hope is to provide the Stratus solution to future groups to eliminate the design trial and error process.

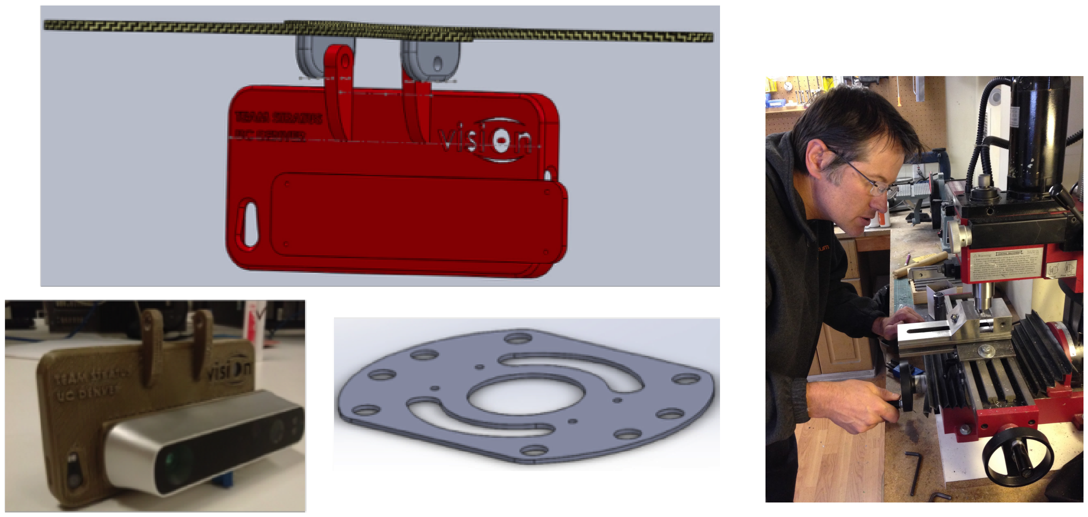


Figure 7: Stratus Prototyping Design Process with Solidworks and Machining Custom Components

Key Features

The Stratus design can be divided into four functional subsystems as shown in Figure 8. These systems are integrated to allow multiple streams of communication between the robot and control platform. The specific subsystems consist of both software and hardware. Further technical details of each subsystems key features are described.

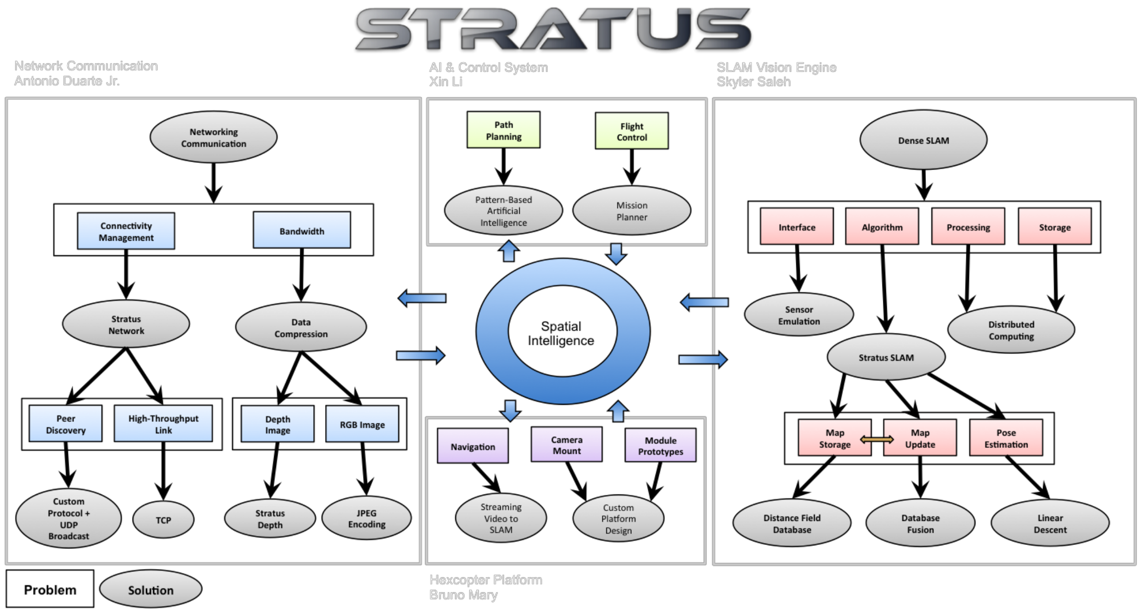


Figure 8: Stratus Integrated Functions of Vision, Communication, Control, and Platform

Low Latency Wireless Video Streaming

- QVGA Depth Camera
- QVGA Color or Night Vision Camera
- 30FPS

Dense Simultaneous Localization and Mapping

- Low Drift 6-axis absolute pose estimation (x,y,z,yaw,pitch,roll)
- High Accuracy 6-axis velocity measurements(translational and rotational)
- Dynamically Builds Dense 3D Scan of Environment
- Large map size only limited by available hard drive space

Sensor Emulation

- Emulate an unlimited amount of virtual sensors using the dynamic SLAM map
- Lidar
- Range Finders
- Cameras

Networking

- Specialized network protocol optimized for robotics
- Networked modules automatically configure and connect to each other
- Effortless distributed computing
- Designed for low latency operation

High Level Robot Control

- Integrates with Existing Robotic Platforms controlled by:
    - Mission Planner
    - Arduino
- Provides RC Control of robots through common PC video game controllers
- Generalized robot control interface
- Includes basic robot behaviors:
    - Land/Launch
    - Obstacle Avoidance
    - Perimeter Following

Wireless Hardware Interfaces

- UART
- I2C
- SPI
- PWM
- GPIO

Test Infrastructure

- Wireless deep robot state inspection
- Wireless debug console interfaces
- Allows online reconfiguration of entire robot pipeline without wiring changes or recompilation
- Network data recorders and playbacks for automatic test vector generation
- Robot Telemetry Logging

Power

- Built in rechargeable 2200mAH battery for powering module and connected usb devices
- 7.2V power output capable of 1.5A for powering sensors and small robots.
- Self contained, no external power required (module).

Product Performance Evaluation

There are multiple ways to evaluate the Stratus product performance. The three core evaluation metrics relate to the Stratus Vision, Stratus Network, and Stratus Control & Navigation. Each area has specific evaluation metrics.

Stratus Vision

There are two principles of the Stratus SLAM implementation to be evaluated. The first is *Pose Estimation*, which estimates the pose of a camera inside of the currently built map, based on what it sees. The second is *Database Fusion*, which builds a map given what the camera sees and the cameras estimated position. These modules are connected by an iterative process with many feedback loops.

Figure 9 shows the data flow diagram for the standard database fusion stage. It shows the input received from the previous stage (pose estimation), and it shows the actual output of this stage, the 3D World Map, which contains the scan of the scene.

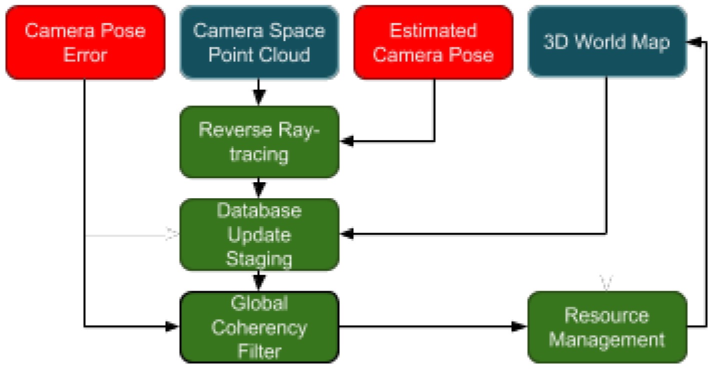


Figure 9: Stratus Database Fusion Data Flow

The pose estimation is an iterative process that refines a guess for the location of the camera in the scene. It works by linearizing the error function to calculate the gradient of it for a given pose. Then a steepest descent solver is used to locate the pose for which the gradient is zero.

The gradient of the error function is highly parallelize able, since it is the sum of many different terms as shown below.

$$\nabla E(\{p_k\}) = \sum_k \nabla \psi(p_k) \psi_\sigma(p_k) \sigma_k$$

The database fusion stage updates the world map with a new data. It is only updated once a pose with high confidence has been found. The stage consists of several sub stages that are designed to maximize the reliability of the data stored in it.

The first stage in the process, reverse ray-tracing, works by tracing a path from each measured point back to the camera. Every voxel gets an update staged for it. This allows the system to utilize what the camera doesn't see to improve the database. All updates are then staged until a certain confidence level is achieved, as dictated by the global coherency filter. Once the updates reach high confidence, the samples are passed to the resource management stage, which performs the required steps to update the database. Either allocating, freeing, or replacing existing samples in the map.

The full SLAM pipeline is just a repetition of the the Pose Estimation and Database Fusion Stage, based on the best estimated pose.

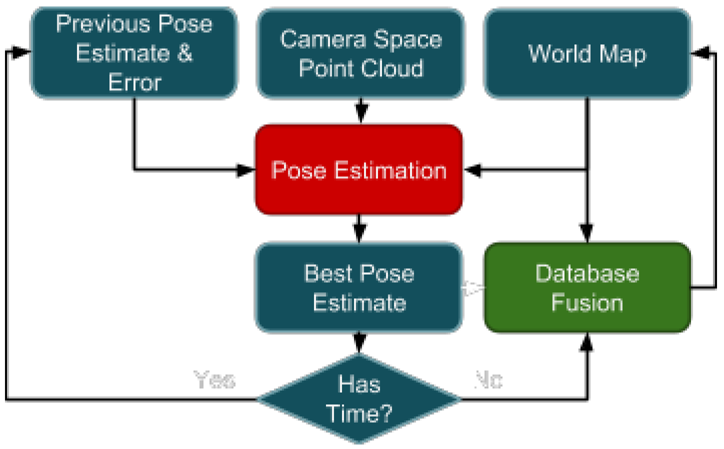


Figure 10: SLAM Overall Flow Control

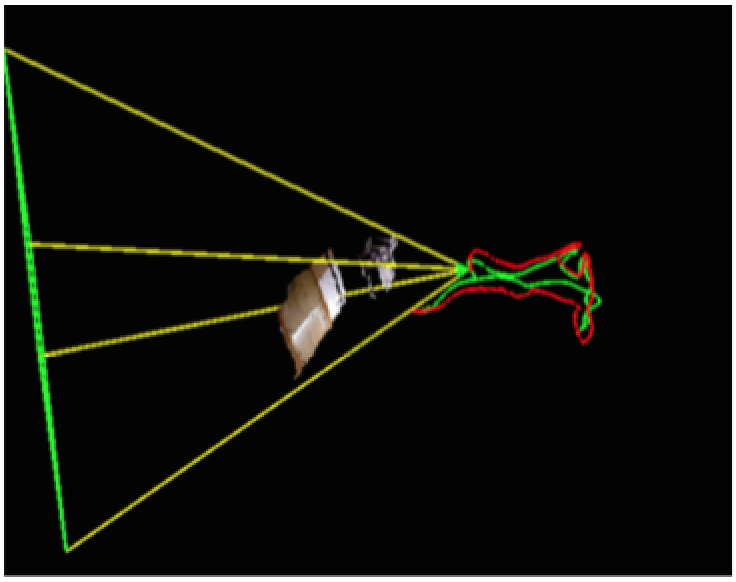


Figure 11: Example Stratus SLAM (Red is the calculated path, Green is the true path)

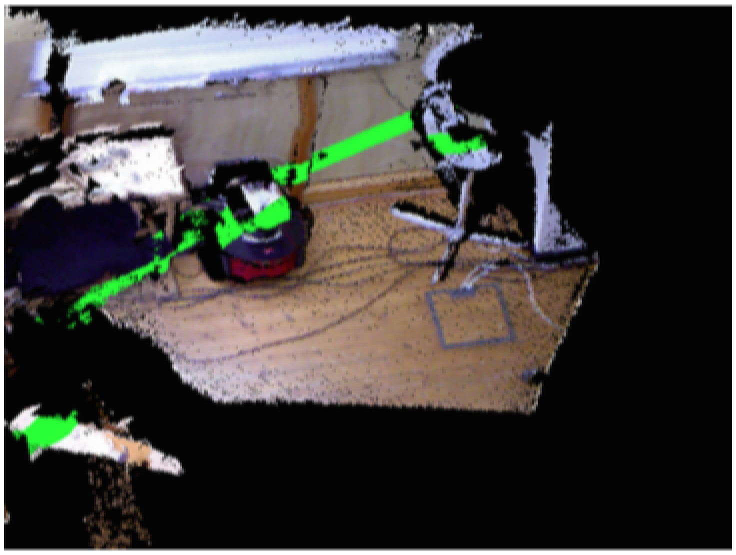


Figure 12: Example Stratus SLAM Scan of Area

Stratus Networking

There are different ways to measure the performance of a network, as each network is different in nature and design. The following measures are often considered important and evaluated for StratusNet:

- Bandwidth: commonly measured in bits/second is the maximum rate that information can be transferred.
- Throughput: actual rate that information is transferred.
- Latency: delay between the sender and receiver decoding it, this is mainly a function of the signals travel time, and processing time at any nodes the information traverses.
- Range: limited range (distance) by the transmission power, antenna type, the location, and the environment.

The available channel bandwidth and achievable signal-to-noise ratio determine the maximum possible throughput, which is not generally possible to send more data than dictated by the Shannon-Hartley Theorem. The necessary bandwidth for the StratusNet relates to the biggest data transmitted through the network, in which is the video streaming. The color and depth image transmitted in 30 frames per second, after compressions, requires approximately 10 Megabits per second of bandwidth. Since the images are streamed to the computer to be processed, the easiest way to measure the bandwidth and monitor the throughput is using a software (Activity Monitor on Apple Mac Pro computer) that allows to visualize the data rate coming to the client.

Latency is a time interval between the stimulation and response. Network latency can be measure either one-way, the time from the source sending a packet to the destination receiving it, or round-trip delay time, the one-way latency from source to destination plus the one-way latency from the destination back to the source. Round-trip latency is more often quoted, because it can be measured from a single point. Round-trip is the way latency is evaluated into StratusNet, in which achieves less than 20 milliseconds, allowing video streaming without human delay perception.

Building a network to communicates robots that can fly or go through the ground requires the measure of range, since modules can lost connection being too far away and the system needs to be aware of that. When the robot is used inside, walls and other obstructions can also be constraints. Consumer wireless routers vary in the Wi-Fi range they support. Routers with stronger Wi-Fi signals allow devices to connect at higher speeds from a greater distance and stay connected more reliably. A wireless router's antenna technology generally determines its Wi-Fi signal strength and hence its range. For example, 802.11g (StratusNet Wi-Fi) wireless routers offered better Wi-Fi range than comparable 802.11b units due in part to technical improvements in their antennas. The StratusNet router specifications affirms that it works well for 200 ft range, but practical tests, mainly flying the Stratuscopter, have shown that a 100 ft range is a more confident metric.

Stratus Control and Navigation

Team Stratus conducted integration platform control tests at the Cherry Creek State Park flying field shown in Figure 13. Team Stratus interfaced the custom controls with the APM flying software via Mission Planner. Simplification of a user interface, where any user will be successful in controlling the flying robot, was the team's primary focus.

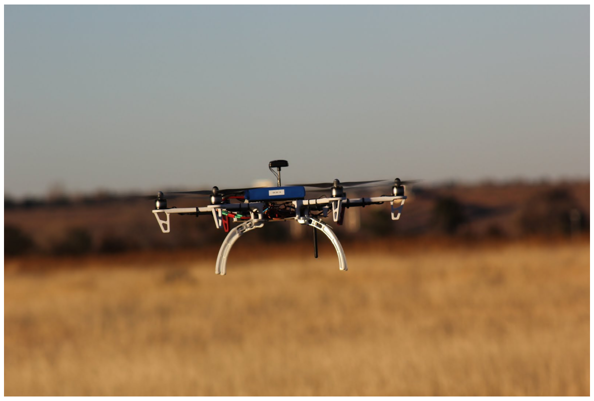


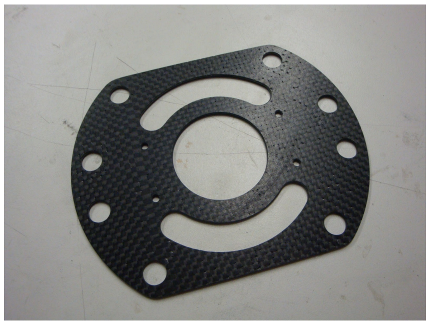
Figure 13: Stratus Testing at Cherry Creek State Park

Team Stratus's objective was to obtain a low latency that would diminish the negative impact that could occur on the platform, such as a lag in response time between user and drone.

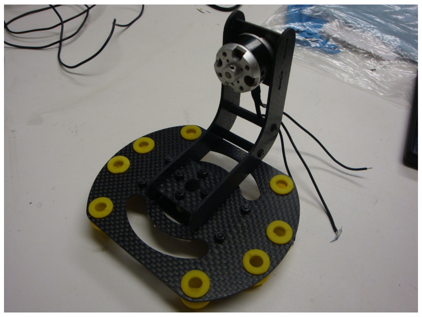
Technical Specs:

- A DJI Flame Wheel F550 (drone) was selected for the project so as to obtain a mainstream platform that integrated the navigation module with the platform.
- The Pixhawk Flight Controller was selected and installed for its ability to utilize open-source software such as APM.
- A custom mount was designed and manufactured by Team Stratus to accommodate the navigational/visual module. Carbon fiber and Solidworks 3-D software was utilized for this purpose. See photos below:

"Custom Mounting Plate"



"Custom Assembly"



Network Architecture

The capabilities of modern robotics are directly limited by the amount computational power that they can carry. To extend the capabilities, the Stratus framework enables a networking layer that can transmit multiple formats and amounts of data from sensors and cameras to remote operating stations. The custom networking solution is called StratusNet designed as a transport-layer that transmits data over a standard WiFi (IEEE 802.11g) connection. A component of the network architecture is the ability uses a set of integrated compression algorithms to reduce the communication latency, specifically more efficient 3D depth sensing and video transmission.

StratusNet is inspired by Internet of Things (IoT) concept and has a custom network architecture designed for both the Transmission Control Protocol (TCP) and User Datagram Protocol (UDP). The goal of leveraging IoT concepts was to enable modules to auto-configure and connect. The server (Figure 14) uses UDP protocol to broadcasts peer information to any client connected to the same wireless, and makes no need of IP and port setup every time a new testbench is necessary. As soon as the client finds a peer and connects to the server (Figure 15), the TCP protocol guarantee reliable connection and streaming of data, like video streaming.

The UDP is used for established the connection between any modules. One module attempting to connect broadcasts peer informations to any other module connected into the same wireless using UDP. But, the UDP protocol does not guarantee the delivery of data. Thus, with the necessary information, the TCP protocol enables two modules to establish connection used to guarantee the delivery of data and also guarantees that packets will be delivered in the same order in which they were sent.

The IoT idea proposes the development of a network in which everyday objects have connectivity, allowing them to send and receive data without requiring human-to-human or human-to-computer interaction. The StratusNet is inspired on it auto-connecting modules (robot, computer, cellphone), which is made through UDP protocol broadcasting/discovering peer information. Thus, robots can make use of it to easily connect to each other, or to connect to a remote station, like a computer. The Stratus project handles many modules connected through the wireless, so the process makes test benches much easier and faster, with 'automatically' setup (no need port and IP setting).

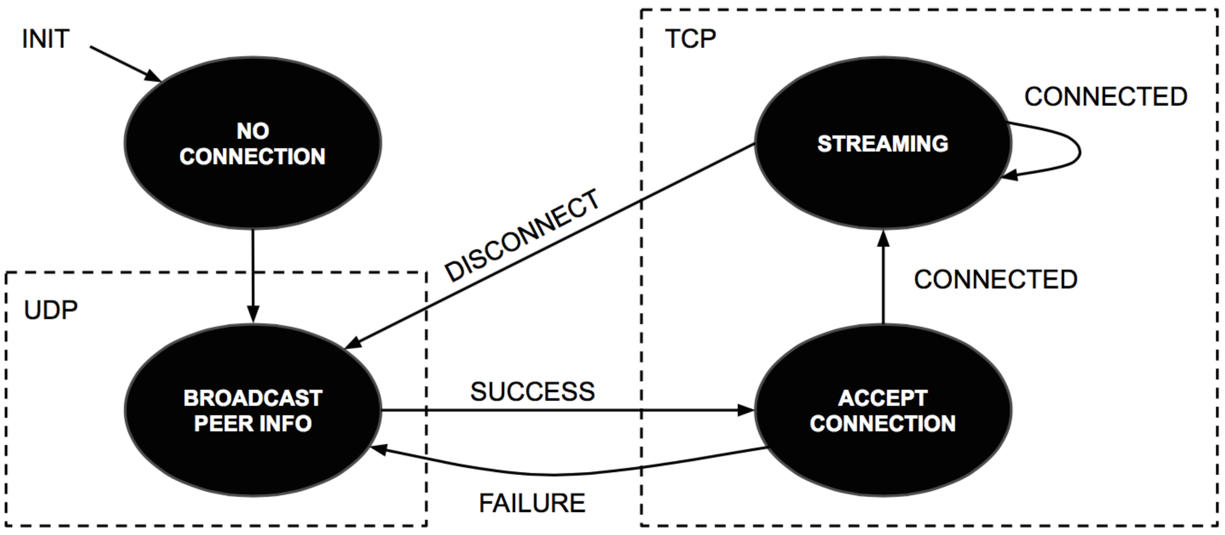


Figure 14: Server Diagram

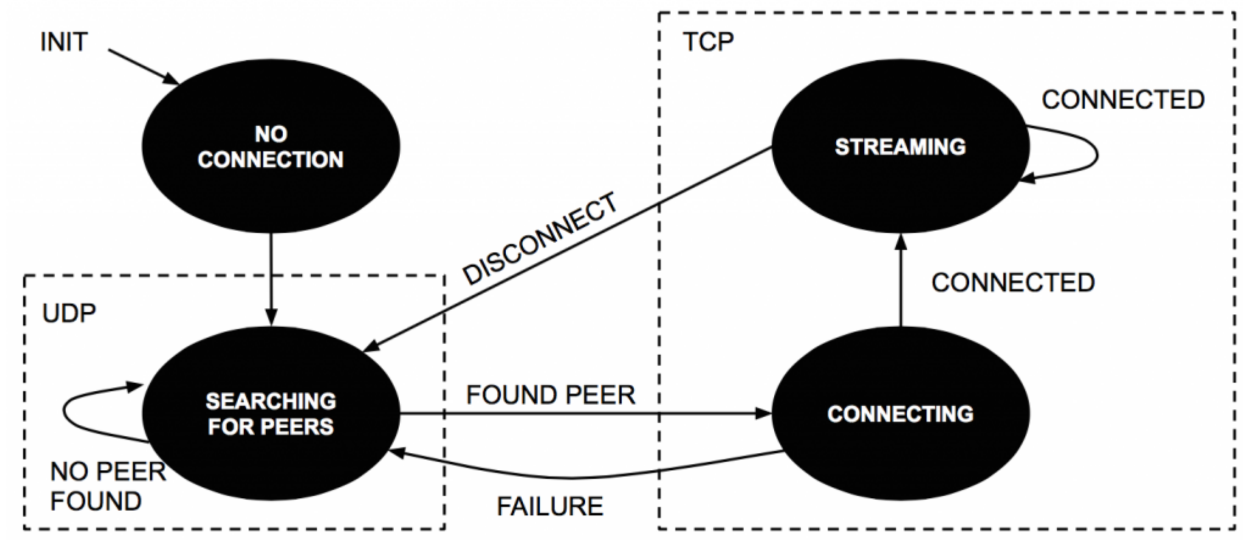


Figure 15: Client Diagram

Versatile Development

As developers of robotic solutions deploy code in different languages and have different data types to transmit, a primary goal for StratusNet was to create a versatile framework. As such, the StratusNet network library modules were constructed using both C/C++ programming and Python modules. The dual-language network library serves well for accommodating different users as well as multiple simultaneous robot platforms. To additionally improve the assistance to other robotic projects, special attention was dedicated to lowering network latency of high-bandwidth and critical data such as image streams, 3D sensor maps, and object avoidance commands. To provide versatile development, rather than have fixed port listing for each function, all devices use a port name space to help eliminate the programming burden of fixed port positions. The Stratus ports are listed below.

Stratus Port Listing

Function	Port
Compressed Video Stream	video.stream
SLAM Virtual Lidar	slam.virtual.lidar
SLAM Pose	slam.pose
SLAM Virtual Camera	slam.virtual.cam
Joystick Control	joy.input
Rover Control	joy.rover
Mission Planner Control	joy.mp
Raw Video Stream	video.raw

Stratus Module

Mechanical Components

The ground module is equipped with an external depth sensor camera while a power source and an Edison powered board, which is the onboard processing and communication device, are combined in a custom protective enclosure. The air module consist of a depth sensor camera, an Edison powered board, and a programmable voltage controller which are conveniently assembled in a carbon fiber custom enclosure. The power source is provided externally by the flying robot main power supply. The air module can be adapted to a stabilization system for smoother scan.

Function Description

The Stratus module is the center of the robot add-on navigation hardware, it’s function consist of acquiring visual data and communicate to the ground unit for processing.

Wiring Diagram

Wiring from the Edison to the robot is done by using an L shaped SATA connector (Figure 16). V7 is the power pin, BAL pin controls the battery balance since the Stratus Module is powered by two 3.7V batteries in series. RX/TX pins are for data transmission.

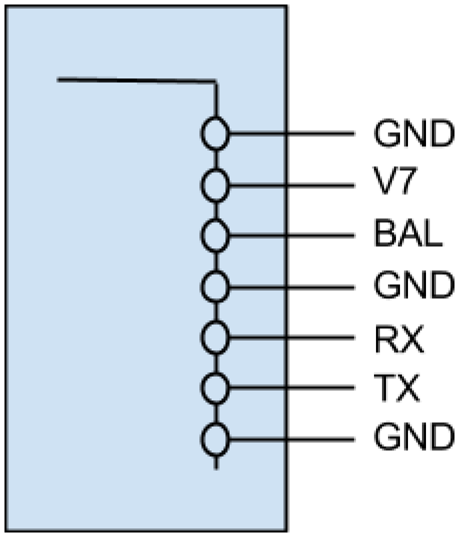


Figure 16: L Shaped SATA Connector

Joystick Control

- Joysticks - In order to control many modules of this project with various functionalities, using a joystick would be the easiest approach and Stratus used a PS3 controller (Fig 17) for its built in bluetooth and its compatibilities with the ground station OS. A PS3 controller would be way easier for a person with no flying experiences to control since it has a lot less inputs which in turn would be easier to use, comparing to a radio transmission controller (Figure 18) for a typical flying platform with large amount of inputs and is usually intimidating for first time users.
- StratusNet - Using StratusNet, inputs from a joystick such as the PS3 controller will be sent directly to the ground station. Then it will be mapped to individual commands to control the platforms.

Figure 17: Radio Transmission Controller

Figure 18: PS3 Controller

- Libraries - Stratus utilized many useful functions from GLFW and OpenCV. GLFW was used for the joystick control by mapping the control input to the PS3 controller so that controlling the platform manually as an alternative to the script was an option.
- It is important to see the results of the joystick inputs so Stratus wrote the Control Center as a visual representation of the the modules together. As a part of the Control Center, the Joystick Control (Figure 19) demonstrates directly on which button is being pressed on the joystick.

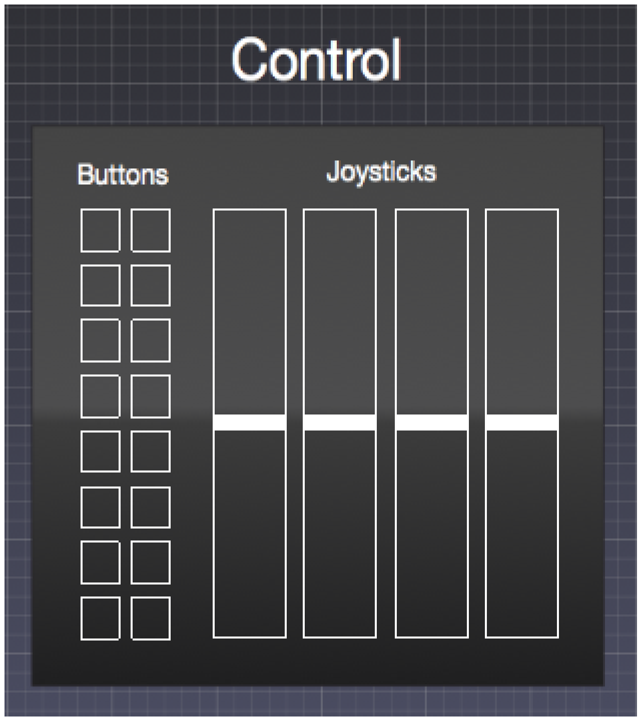


Figure 19: Joystick Control Interface

Figure 20 illustrates the full Stratus control center that provides in real-time a display of the video streaming, depth map, dense SLAM of the robot's sensing array as well as the controller status, robot velocity and orientation, and the camera pose (trajectory).

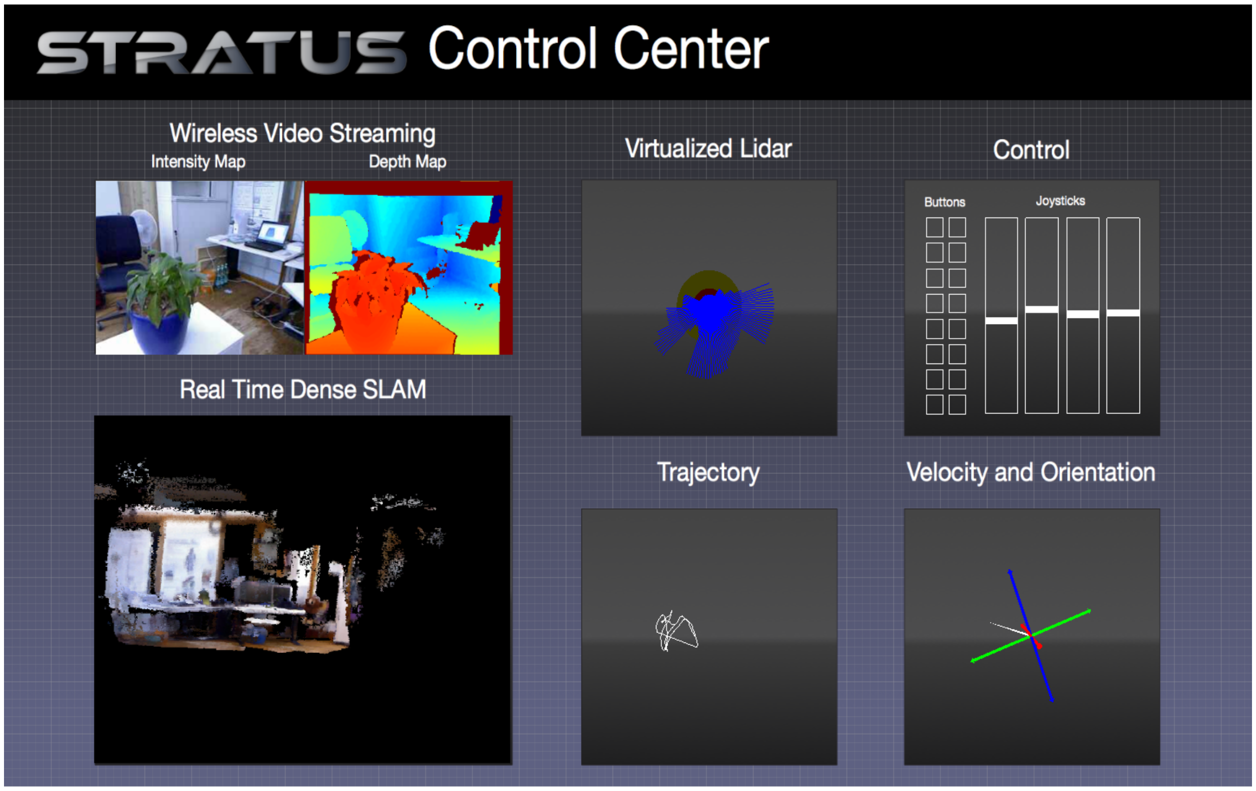
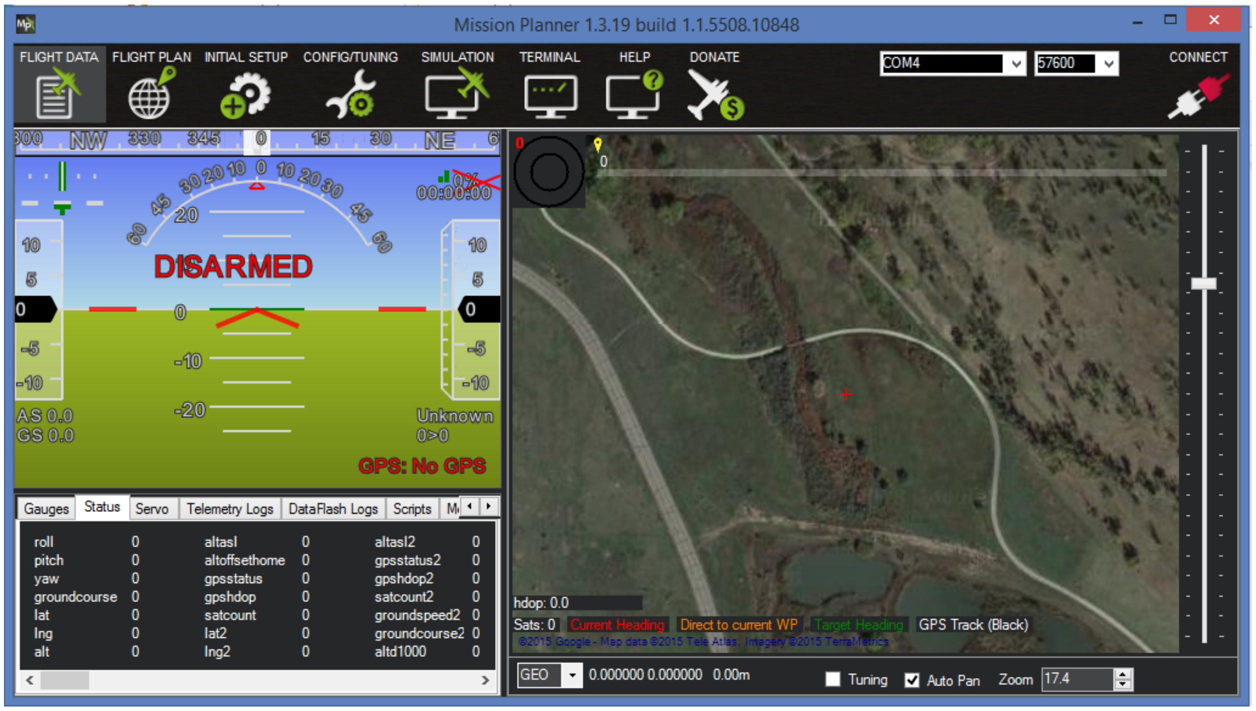


Figure 20: Full Stratus Control Center

Rover Interface

Mission Planner Interface

- Mission Planner is generally used for radio transmission and calibration of the platform. Stratus chose Mission Planner as the flight control interface because it is opensource and it works right out of the box which saves a lot of time since Stratus does not have to rewrite the interface from scratch
- Stratus platform uses the Mission Planner Interface (Figure 21) as its flight control. Stratus is able to interface with Mission Planner by using python script, which is the default script Mission Planner takes in, to send commands directly to the flight control for the navigation.
- Mission planner also has a built in GPS which Stratus plans to use for the outdoors as an alternative navigation method, and switch to Stratus Vision as soon as the system is in doors.



COM4 57600

roll	0	altasl	0	altasl2	0
pitch	0	altoffsethome	0	gpsstatus2	0
yaw	0	gpsstatus	0	gpshdop2	0
groundcourse	0	gpshdop	0	satcount2	0
lat	0	satcount	0	groundspeed2	0
lng	0	lat2	0	groundcourse2	0
alt	0	lng2	0	altd1000	0

hdop: 0.0
Sats: 0

GEO 0.000000 0.000000 0.00m ☐ Tuning ☑ Auto Pan Zoom 17.4

Figure 21: Mission Planner Interface

Video Streaming

The Stratus video streaming allows the camera to be view from ground, which enables algorithms to run into the remote station to use the computer power. The streaming also can be useful to send additional flight informations, such as battery level, emergency situations, and camera calibration parameters. Images are the color image and depth image, both 320 x 240 frame size and are streamed in 30 frames per second (Figure 22). The Stratus network, like any wireless network, has limited bandwidth and range, thus different images compression techniques were performed in order to minimize the image size as much as possible to make streaming more reliable.

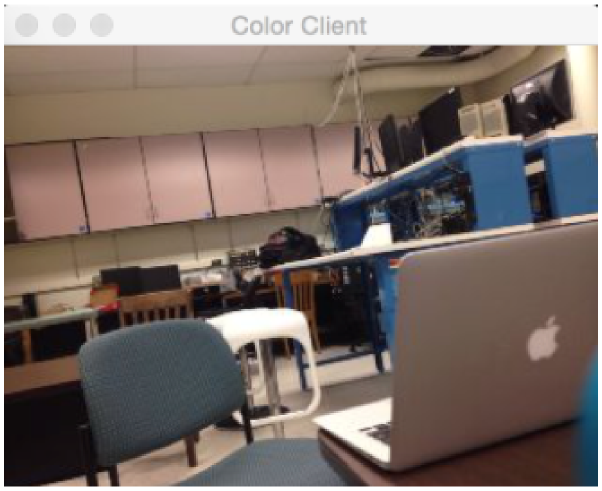


Figure 22: Color Image

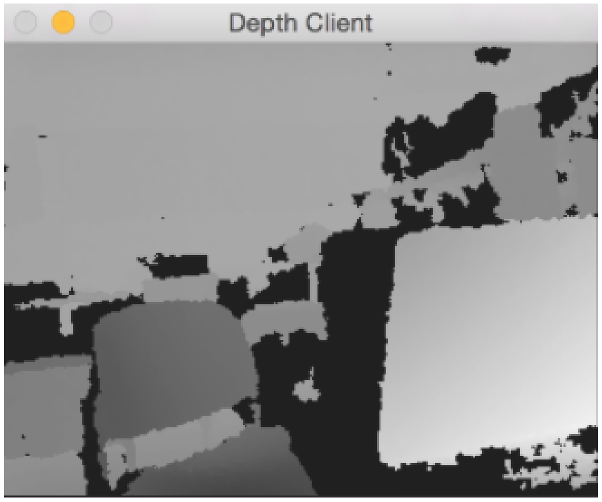


Figure 23: Depth Image

Figure 24 compares the initial and after compression transmission rate. As shown, the original data needs about 90 Megabits per second, and with the RGB565 color encoding approach it becomes 80% of the original data. While the standard JPEG encoding brings the rate to almost 40 Megabits per second, which makes it a better approach for color image compressing. Depth image compression was made using the depth image characteristics of repeating values, the sensor accuracy, a logarithmic transformation and the run-length encoding, named Stratus Depth Compression.

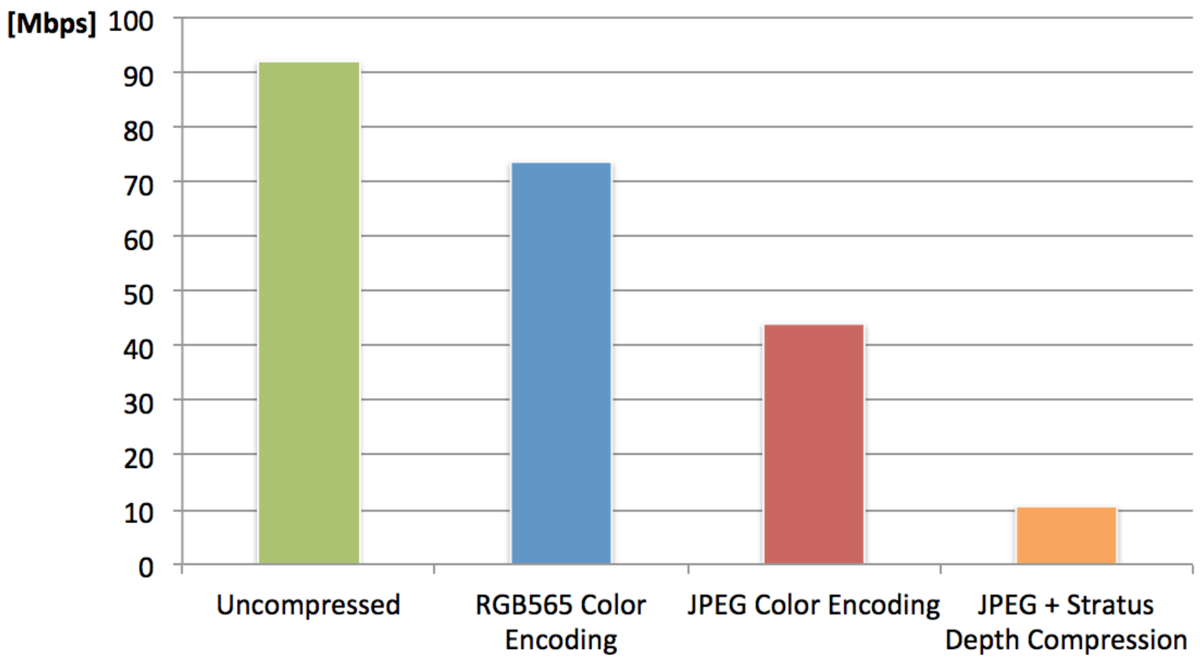


Figure 24: Video Compression Approaches for Transmitting Image Frame

For Stratus Depth Compression the Figure 25 illustrates the linear map of the sensor accuracy showing that it has not enough precision for close objects and has too much precision for far objects, which requires a large numeral range to store all values.

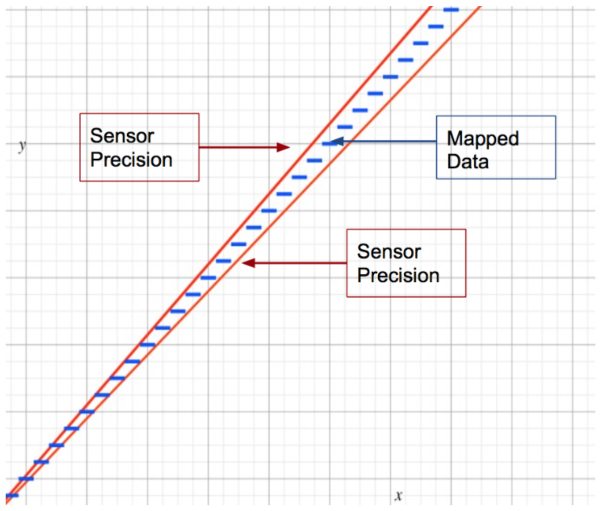


Figure 25: Linear Mapping

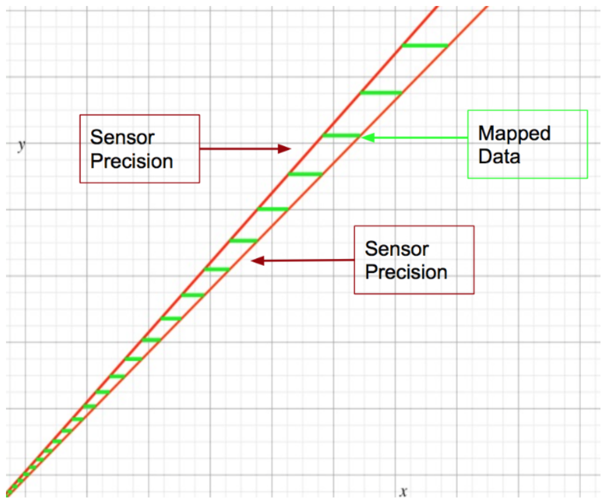


Figure 26: Logarithmic Mapping

Performing the logarithmic transformation makes mapped data (Figure 26) to use the sensor precision more accurately both for close and far objects and reduces the required numerical range considerably. After the logarithmic transformation we perform the run-length encoding, which is a very simple form of data compression in which sequences of data, where the same data value occurs in many consecutive data elements, are stored as a single data value and count, rather than as the original repeating values. The final result including color and depth compression and its comparison to previous approaches (color image compressions only) are shown in Figure 24 and brings the final data rate to almost 10 Megabits per second (Mbps), which combined with the networking allows the streaming of images with no perceptive delay.

The Stratus video streaming technology includes one recording library to store streamed images, allowing re-running the SLAM algorithm and building the 3D map for the same set of images.

Pattern Based Navigation

To achieve navigation, specifically obstacle avoidance, Stratus used a pattern based behavior to control the robot. By finding certain patterns from the maps generated by StratusVision, Stratus is able to match certain cases that decides what the robot should do.

- Network interface - Using StratusNet, information will be sent from one machine (server) to another machine (client). Stratus use computers as ground stations to compute the necessary behaviors from the generated map for processing. Then send data to the client. Tests were done on a rover. With this kind of distributed computing, microcontroller on the rover will only be controlling the speed of the motors.

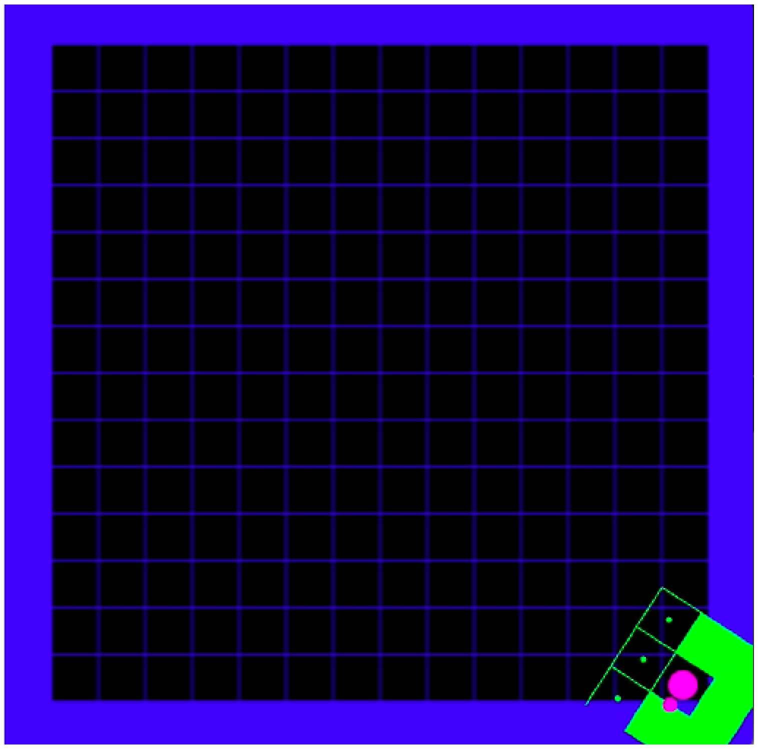


Figure 25: World Grid & Robot Grid

- Pattern Diagrams - By pattern matching, Stratus is able to make different logic cases to determine the behavior of the platform. The idea is to have a world grid and a robot grid so we can match patterns (Figure 25). Stratus determined it would be more convenient to use a pi grid instead of a square grid and instead of the world grid patterns of the map generated from SLAM and calculating the distance between the sensor and the object.

- Rationale - Stratus wanted to make the logic of pattern based behavior as simple as possible, enough for the platforms to travel around a basic layout.

Project Execution Performance Evaluation

Time Line

- August, 2014 - Started Research
- September, 2014 - Decided that we were going to research SLAM, read some papers
- October, 2014 - Xin started to research SLAM, Skyler working on his own SLAM. Bruno joined the team, we now have a hexacopter, so we decided that we can use it as a testing platform
- November, 2014 - Came up with some ideas on how exactly we are going to execute our ideas. Antonio joined team and start to develope the StratusNet, Skyler finished SLAM, started to work on camera prototype 1
- December, 2014 - Each team members worked on own module while collaborating on the Cornell Cup presentation. Xin finished robot behavior visuals, Antonio finished StratusNet, we can now start to integrate modules together through it. Also worked on camera prototype 2
- January, 2015 - A lot of Hexacopter testings, started on some of the platform scripting, mainly testing SLAM and capturing footages of drone flying
- February, 2015 - Polished up some of the modules, getting ideas and working on camera prototype 2, some more platform testing. Got joystick to work with the rover test platform
- March, 2015 - Worked on mission planner and Python script in order to get the platform working through scripting, some more joystick stuff and StratusNet polishing. Worked on rover platforms along with joystick. Wrote Stratus Control Center to display important information
- April, 2015 - Antonio working on porting StratusNet with python so we can do stuff with the scripting. Getting videos and data to show up on the visual poster

Budget

Stratus Rover Module(2)

Item	Quantity	Price
Intel Edison + Mini breakout Board	2	$109.90
Battery	2	$52.00
Structure Sensor	2	$758
Toggle Switch	2	$2.58
Case	2	$26.89
Female-Female Dupont Ribbon Cable	1	$1.00
TOTAL	**2 units**	**$950.37**

Stratus Copter Module

Item	Quantity	Price
Intel Edison + Mini breakout Board	1	$54.95
Module mount	1	$50.00
Module stabilization motor	1	$19.99
Module stabilization controller	1	$58.10
Structure Sensor	1	$379
Case	1	$20.00
Female-Female Dupont Ribbon Cable	1/2	$1.00
Toggle switch	1	$1.29
Miscellaneous	1	$18.64
TOTAL	**1 unit**	**$602.97**

Figure 26 shows the full integration of components for the Stratus Hexacopter design.

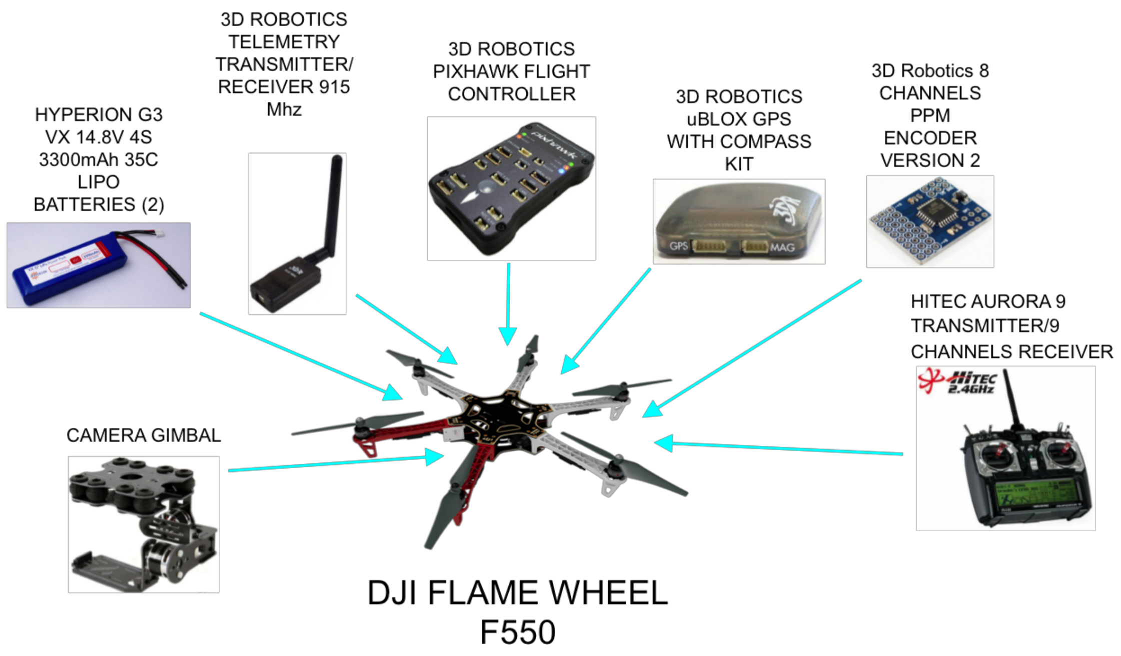

HYPERION G3 VX 14.8V 4S 3300mAh 35C LIPO BATTERIES (2)
3D ROBOTICS TELEMETRY TRANSMITTER/ RECEIVER 915 Mhz
3D ROBOTICS PIXHAWK FLIGHT CONTROLLER
3D ROBOTICS uBLOX GPS WITH COMPASS KIT
3D Robotics 8 CHANNELS PPM ENCODER VERSION 2
HITEC AURORA 9 TRANSMITTER/9 CHANNELS RECEIVER
CAMERA GIMBAL
DJI FLAME WHEEL F550

Figure 26: Hexacopter Assembly of Components

Stratus Copter (for Testing)

Item	Quantity	Price
DJI Flamewheel F550 ARF	1	$343.31
3D Robotics Pixhawk flight controler	1	$269.04
3D Robotics transmitter/receiver	1	$127.68
PWM encoder	1	$27.22
External 3D Robotics compass/GPS	1	$107.61
Hitec transmitter/receiver	1	$349.99
Landing gear	1	$18.15
range finder	1	$46.10
battery ,(Hyperion, 14.8V, 3300 mAh, 35C)	2	$149.90
Hitec multi charger	1	$129.99
Power Supply	1	$99.99
TOTAL	**1 unit**	**$1,668.98**

Stratus Rover (for Testing)

Item	Quantity	Price
Pololu Zumo Kit	2	$199.90
Arduino Leonardo	2	$49.90
0.1” 20 Pin Headers	2	$2.00
TOTAL	**2 units**	**$251.80**

Funding and other resources

Source	Amount
U.R.O.P	$2,400
Cornell Cup	$1,500
Team Personal Investment	$1,541.30

Hardware

Three Intel-based processing components were used throughout the design. The core Stratus module uses the Intel® Edison Board shown in Figure 27 with the following capabilities:

Intel® Edison Board

- Intel® Atom™ system-on-a-chip (SoC) based on leading-edge 22 nm Silvermont microarchitecture including a dual-core CPU and single core microcontroller (MCU)
- Integrated Wi-Fi, Bluetooth* LE, memory, and storage
- Support for more than 30 industry-standard I/O interfaces via a 70-pin connector

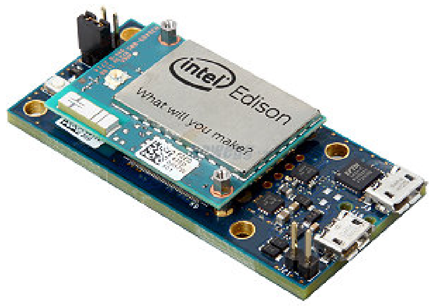


Figure 27: Intel Edison Board

The other two Intel-based processing components involved the processing on the base station laptop. The SLAM codes as well as the Pattern-based navigation run on Intel HD Graphics and Intel Core systems shown in Figure 28.

Figure 28: Hexacopter Assembly of Components

Recommendations and Next Steps

The recommendations for the future work are to complete a prototype of the Stereoscopic (dual camera) module. Figure 29 shows the assembled cameras constructed as a pair. The goal of the stereo camera is to provide a depth estimation based on triangulation that requires the camera intrinsics be measured. In addition, the software support for performing stereoscopic disparity would need to be developed. Based on the support of the OpenCV library for stereo vision, it is estimated that the additional camera module could be integrated into the entire Stratus framework in under 4 weeks of time.

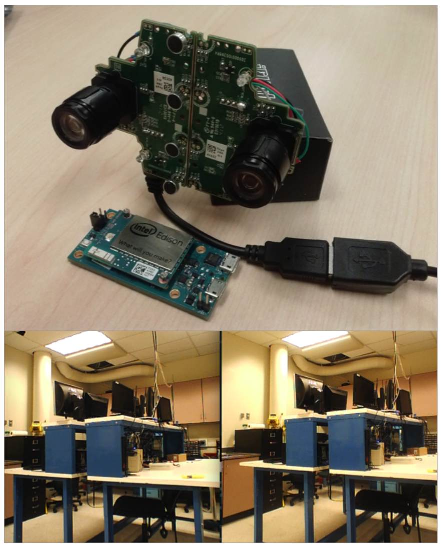


Figure 29: Stratus Stereoscopic Prototype for Depth-Sensing Outdoors with Visual Light

Nomenclature Glossary

COTS - Commercially off the shelf
CPU – Central processing unit
Devboard – Development board
DHCP – Dynamic host configuration protocol
FPGA – Field programmable gate array
GPU – Graphics Processing Unit
IP – Internet Protocol
J – Joules
LAN – Local area network
LCD – Liquid crystal display
LED – Light emitting diode
Linux – An open source, POSIX compliant operating system
Mbps - Megabits per second
OpenCL – Open Computing Language
OpenCV – Open Computer Vision
RAM – Random access memory
RF – Radio frequency
SLAM - Simultaneous Localization and Mapping
SolidWorks – A 3D CAD and rendering software application
TCP - Transmission Control Protocol
Transceiver – Transmitter and receiver
UROP - Undergraduate Research Opportunities Program
UDP - User Datagram Protocol
WAN – Wide area network
WBX - wide bandwidth transceiver

References

http://www.roboticsproceedings.org/rss09/p35.pdf
Dense Visual SLAM for RGB-D Cameras - Computer Vision ...
http://www.graphics.stanford.edu/~niessner/niessner2013hashing.html

Acknowledgements

Faculty and University
Our team would especially like to thank Dr. Dan Connors for recommending the competition and for his relentless effort to ensure we have a successful project. Also, we would like to thank department faculty, Dr. Wendell Chun, for providing valuable insight into the functions of the prototype and several early logistical suggestions. Our team would like to thank the University of Colorado Denver Department of Electrical Engineering for providing initial funding to develop the project and the University of Colorado Denver Undergraduate Research Opportunities Program (UROP).