SpiderFab™: Process for On-Orbit Construction of Kilometer-Scale Apertures

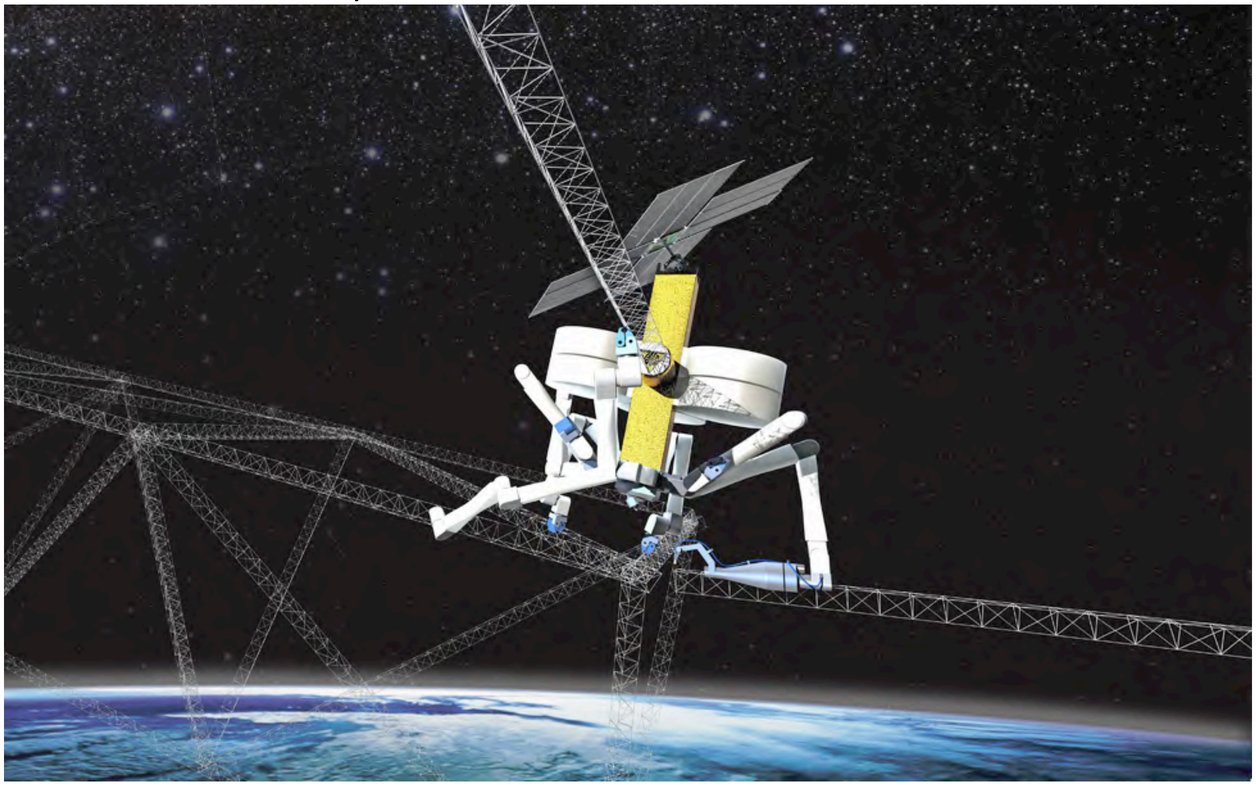
Authors: **Robert Hoyt, Jesse Cushing, Greg Jimmerson, Jeffrey Slostad, Robert Dyer, Steve Alvarado**

Tethers Unlimited, Inc.
11711 N. Creek Pkwy S., Suite D113
Bothell, WA 98011

Final Report
Report Date:
29 February 2016

Period of Performance:
1 Oct 2013 – 30 January 2016

Grant # NNX13AR26G



Sponsored By:

NASA Innovative Advanced Concepts (NIAC)
NASA Goddard Space Flight Center
8800 Greenbelt Road
Greenbelt, MD 20771

SF 298

REPORT DOCUMENTATION PAGE	*Form Approved* *OMB No. 0704-0188*

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE 30 January 2016	83. REPORT TYPE AND DATES COVERED Final Report, 1Oct13 – 30Jan16

4. TITLE AND SUBTITLE **SpiderFab™: Process for On-Orbit Construction of Kilometer-Scale Apertures**	5. FUNDING NUMBERS **NNX13AR26G**
6. AUTHORS Robert Hoyt, Jesse Cushing, Greg Jimmerson, Jeffrey Slostad	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Tethers Unlimited, Inc. 11711 N Creek Pkwy S., D-113 Bothell, WA 98011	8. PERFORMING ORGANIZATION REPORT NUMBER **NNX13AR26G – FINAL**
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) **NASA Innovative Advanced Concepts (NIAC)** **NASA Goddard Space Flight Center** **8800 Greenbelt Road** **Greenbelt, MD 20771**	10. SPONSORING/MONITORING AGENCY REPORT NUMBER
11. SUPPLEMENTARY NOTES	
12a. DISTRIBUTION/AVAILABILITY STATEMENT **Distribution Statement A:** Distribution is Unlimited.	12b. DISTRIBUTION CODE

13. ABSTRACT

The SpiderFab effort investigated the value proposition and technical feasibility of radically changing the way we build and deploy spacecraft by enabling space systems to fabricate and integrate key components on-orbit. In the Phase I we developed a concept architecture for SpiderFab and identified the key enabling technologies required to implement this concept. The Phase II effort focused on developing and demonstrating those key technologies. In the Phase II we developed robotic tools and control software to enable automated assembly of truss structures to support large spacecraft apertures such as antennas, solar arrays, and optics.

14. SUBJECT TERMS			15. NUMBER OF PAGES
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT None

TABLE OF CONTENTS

1. INTRODUCTION

1.1 Summary

The SpiderFab effort has investigated the value proposition and feasibility of radically changing the way we build and deploy spacecraft by enabling space systems to fabricate and integrate key components on-orbit. In this Phase II effort, we have focused on developing and demonstrating tools and processes to enable robotic systems to manufacture and assemble high-performance structural elements that will serve as the support structures for components such as antennas and solar arrays. Through testing of these technologies in the laboratory environment, these efforts have established the technical feasibility of the key capabilities required for in-space manufacture of large apertures such as antennas, solar arrays, and optical systems, maturing prototype technical solutions for these capabilities to TRL-4. The SpiderFab effort has resulted in **successful post-NIAC transition** of the technology, first to SBIR-funded development of a technology for in-space manufacture (ISM) of truss structures, and then to a NASA/STMD-Tipping Point Technologies funded effort to prepare a flight demonstration of ISM of a structure for a GEO communications satellite.

1.2 Background: SpiderFab Phase I Results

1.2.1 SpiderFab Architecture

In the Phase I we developed an architecture for a “SpiderFab” system that integrates additive manufacturing techniques with robotic assembly to enable in-space manufacturing of large apertures. We identified the key capabilities required to implement this architecture and detailed two concept implementations of this architecture, one a mobile robotic system, illustrated in Figure 1, capable of manufacturing spacecraft components such as antenna reflectors, and the second a palletized payload designed to assemble large solar arrays, as illustrated in Figure 2.

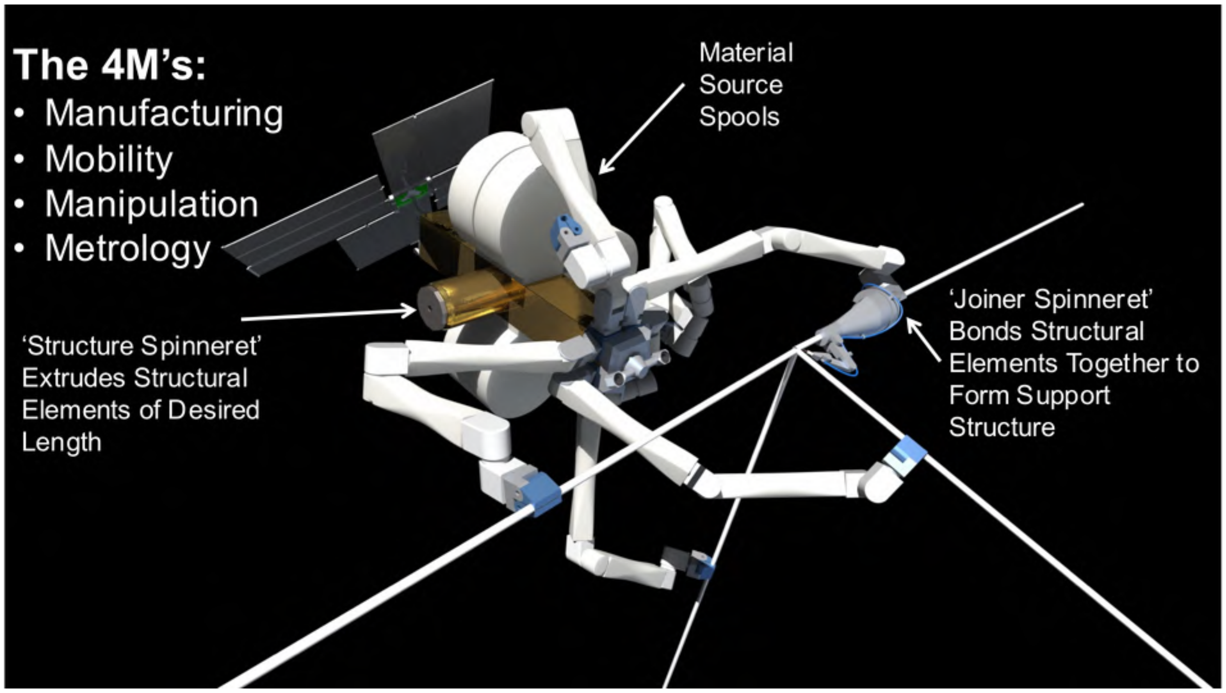

The 4M's:
• Manufacturing
• Mobility
• Manipulation
• Metrology
Material Source Spools
'Structure Spinneret' Extrudes Structural Elements of Desired Length
'Joiner Spinneret' Bonds Structural Elements Together to Form Support Structure

Figure 1. Concept for a ‘SpiderFab Bot’ for in-space manufacture of large support structures.

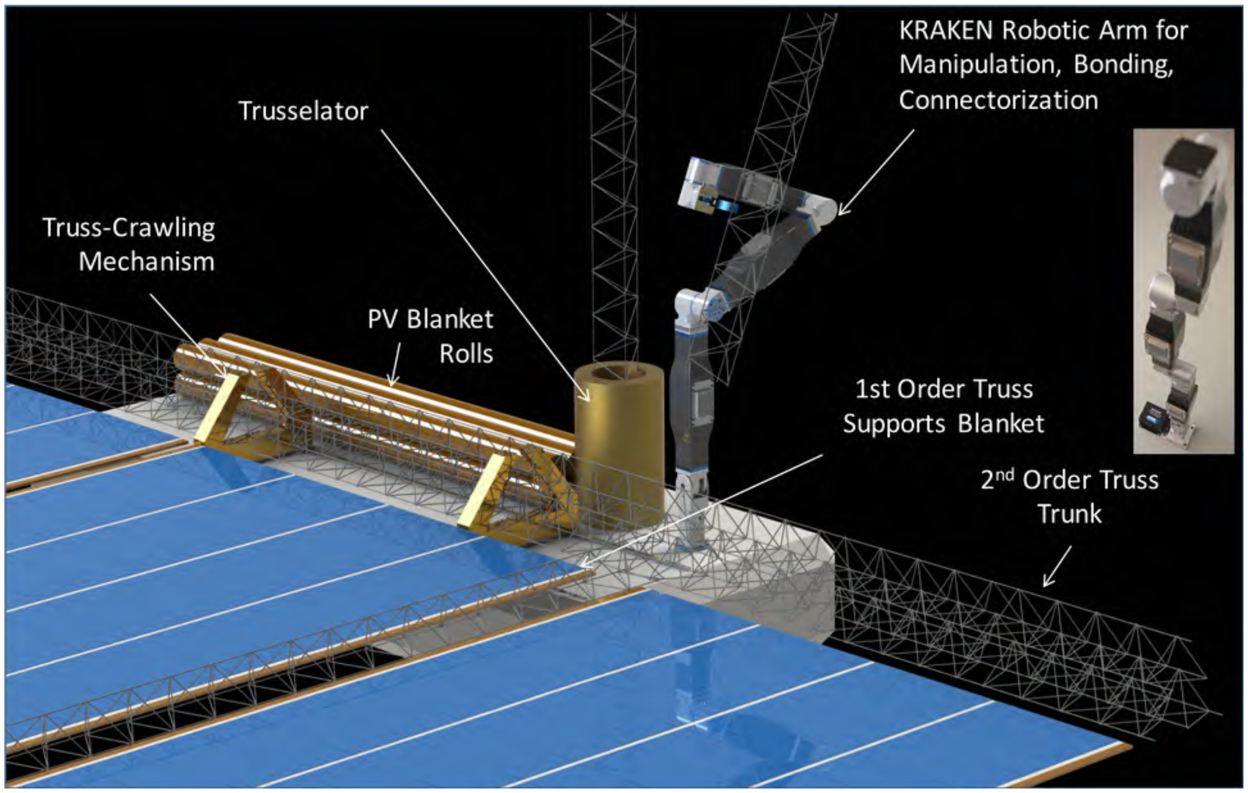

KRAKEN Robotic Arm for Manipulation, Bonding, Connectorization
Trusselator
Truss-Crawling Mechanism
PV Blanket Rolls
1st Order Truss Supports Blanket
2nd Order Truss Trunk

Figure 2. Concept for a palletized SpiderFab payload for in-space manufacture of large solar arrays.

We then investigated the value proposition for on-orbit fabrication of several different kinds of large space system components, and in each case found that the dramatic improvements in structural performance and packing efficiency enabled by on-orbit fabrication can provide order-of-magnitude improvements in key system metrics. For phased-array radars, SpiderFab enables order-of-magnitude increases in gain-per-stowed-volume. For the New Worlds Observer mission, SpiderFab construction of a starshade can provide a ten-fold increase in the number of Earth-like planets discovered per dollar. For communications systems, SpiderFab can change the cost equation for large antenna reflectors, enabling affordable deployment of much larger apertures than feasible with current deployable technologies. To establish the technical feasibility, we identified methods for combining several additive manufacturing techniques with robotic assembly technologies, metrology sensors, and thermal control techniques to provide the capabilities required to implement a SpiderFab system. We performed proof-of-concept level testing of these approaches, in each case demonstrating that the proposed solutions are feasible. These Phase I efforts established the SpiderFab architecture at TRL-3.

1.2.2 SpiderFab Technology Maturation Plan

Figure 3 illustrates an incremental technology maturation plan in which a sequence of flight missions will demonstrate increasingly capable in-space manufacturing solutions, starting with a nanosat-scale demonstration of ISM of a long linear truss for long-baseline sensing applications, progressing to demonstration of ISM of a 2D RF aperture, and then graduating to an operational capability for ISM of very large space systems.

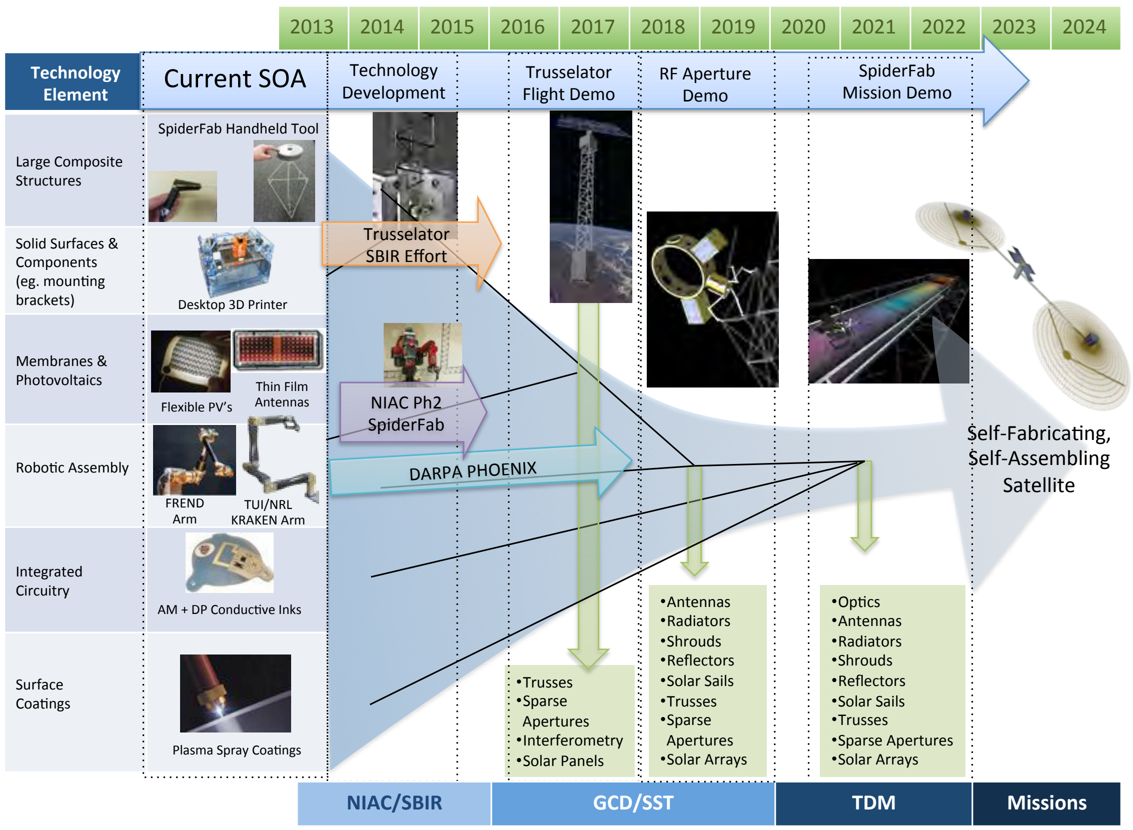


Figure 3. SpiderFab Technology Maturation Plan.

1.3 Background: Trusselator SBIR

In addition to the NIAC SpiderFab effort, TUI is also performing a parallel Phase II SBIR titled "Trusselator", in which we are developing a key initial component of the SpiderFab architecture, a device that converts spools of carbon fiber feedstock into high-performance carbon fiber trusses. The preliminary Trusselator prototype developed in the Phase I SBIR effort is shown in Figure 4 through Figure 7 along with examples of trusses fabricated by the device. Figure 8 shows a 16-m truss sample fabricated with this prototype. This truss sample is light enough yet strong enough to be self-supporting in 1 gee. In the Phase II SBIR effort, TUI is refining the device design to reduce its size, weight, and power (SWaP) and enable it to operate reliably in a vacuum environment. Our goal for this Phase II prototype is to fit the mechanism within a 3U (30x10x10cm) volume to enable affordable flight validation on a CubeSat platform.

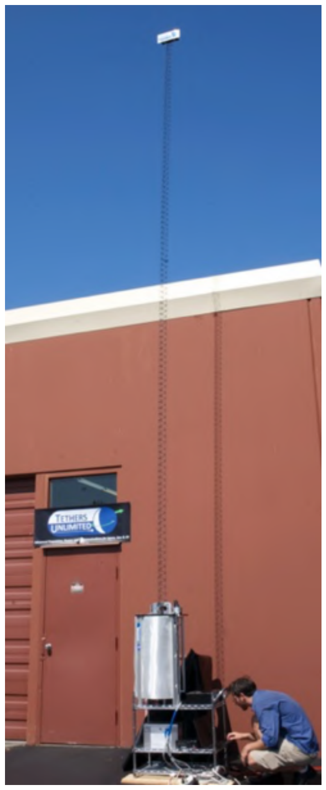


Figure 4. Truss Fabrication demonstration.

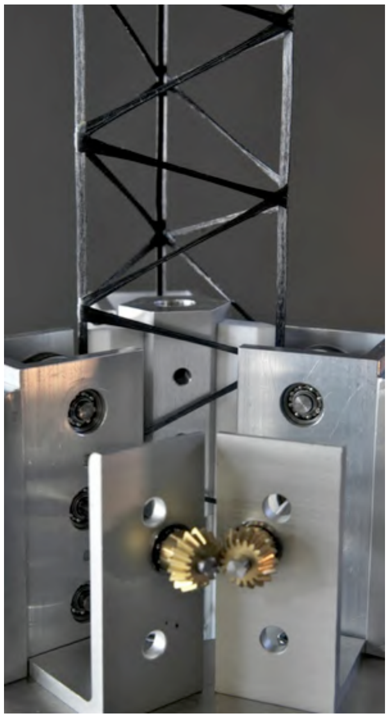


Figure 5. Close-up of the carbon-fiber truss exiting the Trusselator prototype.

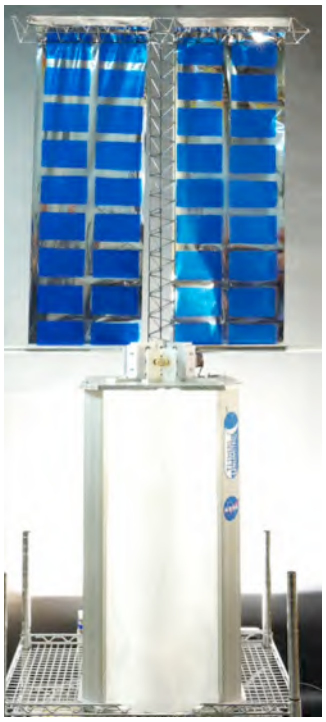


Figure 6. Proof-of-concept demonstration of deploying mock solar panels using the Trusselator.

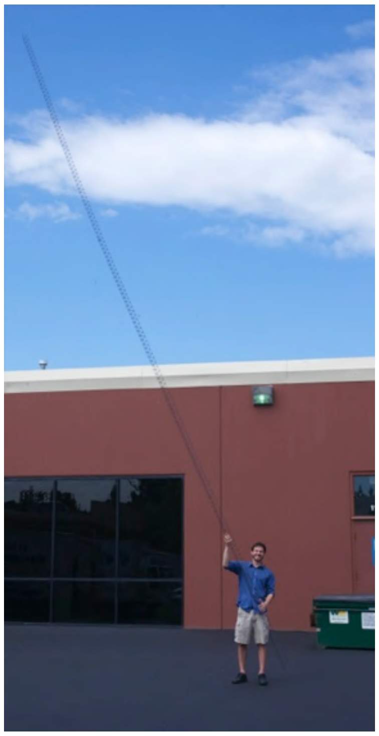


Figure 7. Carbon-Fiber Truss fabricated by the Phase I Trusselator prototype.

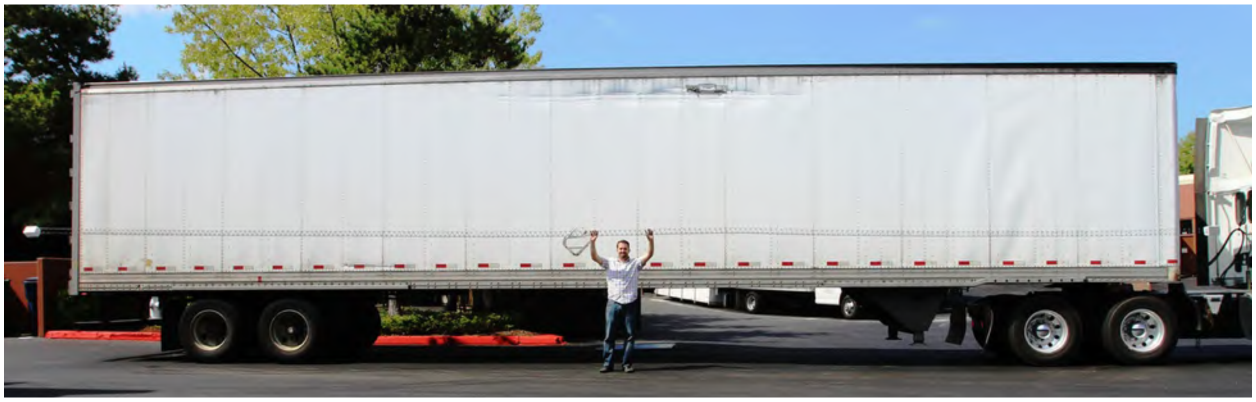


Figure 8. 16m Truss sample, with semi trailer shown for scale.

1.4 SpiderFab Phase II Work Plan

The objective of the Phase II effort was to develop key technologies and mission analyses to mature the SpiderFab architecture to a level where it is suitable for NASA GCD and SST programs to build and fly affordable flight demonstrations. To accomplish this objective, we proposed to (1) design methods to enable in-space manufacture and assembly of structures in the space environment; (2) build and test prototypes implementing these methods in a vacuum environment; (3) develop a concept for a mission to demonstrate in-space manufacture of a large RF aperture; and (4) evaluate the performance, cost, and risk tradeoffs between in-space manufacture and traditional 'deployable' approaches for large apertures.

2. PHASE II ACCOMPLISHMENTS

2.1 Development of Tools for In-Space Manufacture and Assembly of Structures

One of the objectives of this Phase II effort was to design and test methods to enable robotic systems to assemble the 1st-order truss elements fabricated by the Trusselator into 2D and 3D 'truss-of-trusses' structures to create support structures for satellite components such as antennas and arrays. We chose to focus our efforts on enabling creation of such 2nd-order truss structures, rather than simpler approaches such as joining rods together to create a 1st order truss structure (as was illustrated in Figure 1) because 2nd order truss structures can achieve 30X improvements in structural efficiency relative to 1st-order structures.[1] This enhanced structural efficiency is necessary to allow very large (100m-1km) apertures to be built with viable launch masses while achieving the structural stiffness needed to enable attitude and dynamical control of such large structures.

In order to make progress towards the capability to assemble 2nd order truss structures, we designed, prototyped, and demonstrated a SpiderFab "spinneret" tool that a robotic system can use to bond the carbon fiber composite truss segments together. We also prototyped a spinneret tool for free-form extrusion of composite rod segments.

2.1.1 SpiderFab Joiner Spinneret Process Development

To kick off the Phase II effort, TUI performed an in-depth trade study of the key variables that drive the SpiderFab process for fabricating truss-based structures, with the purpose of narrowing down the array of options for development. The complete trade study can be found in Appendix A. The following list of variables was identified, and the noted options selected as being most promising and/or necessary for successful development of the SpiderFab Phase II effort.

A) Feedstock Composition – Carbon Fiber composite with thermoplastic matrix composed of Polyetheretherkeytone (CF/PEEK) was selected as the baseline material for use in development of SpiderFab processes, based upon its combination of high strength, high stiffness, low-outgassing, and high operating temperature capability.

B) Feedstock Format – We selected pre-consolidated CF/PEEK tapes and rods as our feedstock format. This form of feedstock can be stored compactly. Relative to a process that uses separate fiber and thermoplastic as feedstock, using a pre-consolidated composite reduces the mechanical force, power, and system complexity required to achieve high consolidation of the material in the on-orbit processes.

C) Heating Method – We evaluated several different methods for thermally processing and bonding the CF/PEEK materials, including contact heating, ultrasonic welding, and laser heating, and chose contact heating as our baseline approach due to its significantly lower power requirements and lower system complexity.

D) Compaction Method – To ensure high bond strength between joined elements in a structure, we chose mechanical compression techniques.

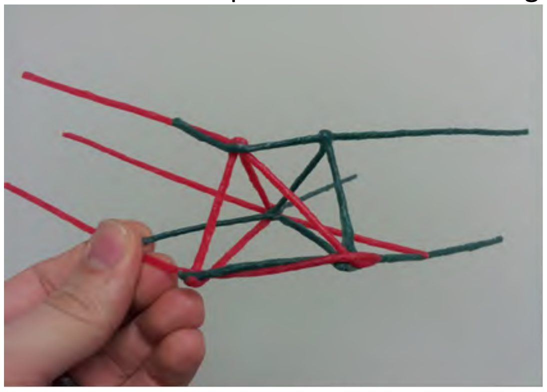
1. Murphey, T.W., Hinkle, J.D., "Some Performance Trends in Hierarchical Truss Structures," AIAA-2--3-1903.

E) Joint Geometry Scheme – To enable high-strength joints between two or more linear truss elements at arbitrary angles, we investigated several concepts, including tapered truss termination and attachment ('point joint'), attachment of directly-contacting existing truss nodes ('butt joint'), and "bridging-the-gap" attachment of non-contacting nodes ('spanner joint'). Based upon our testing with SpiderFab tools, we selected the gap-bridging concept because it achieved high joint strengths with acceptable complexity for forming the joints.

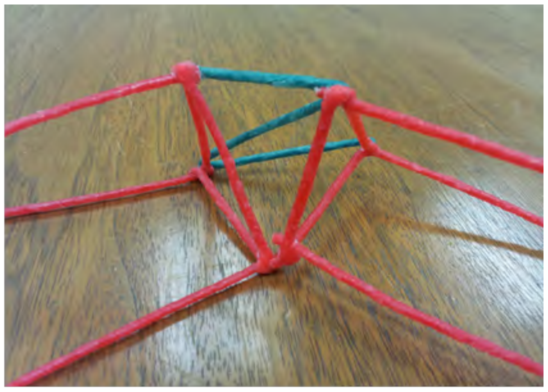
F) Gripping Mechanisms – To enable a robotic system to manipulate truss elements, we developed several gripper mechanism designs to enable precise and repeatable manipulation and positioning of trusses. These end-effector tools must incorporate compliance and/or sensing to prevent damage to the structure.

2.1.1.1 Joint Geometry Development

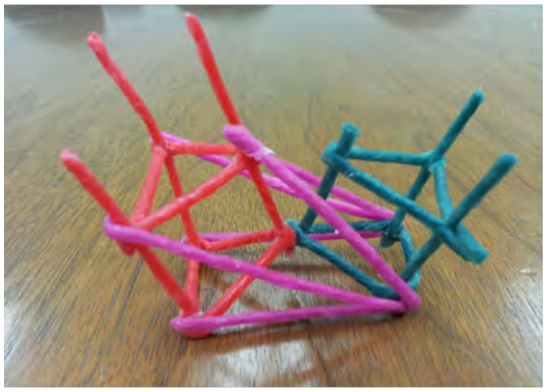
Because the geometry of joints between truss-based structural element is a strong driver of the requirements for a number of the other technical aspects of the assembly process, we investigated several candidate schemes for constructing joints between truss elements. For ease of visualization and manipulation, we used colored flexible composite rods to depict the geometry possibilities, which are many. These colored models were created to simulate some of the recurring types of attachment, including point joints, butt joints, and spanner joints. Figure 9 shows a handful of the possibilities for creating a structure of trusses.



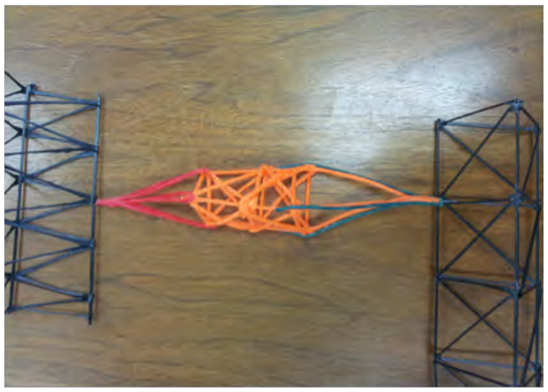
A. Laminate Longeron Excess



B. Join one node directly, add a Z to triangulate the 4 other nodes



C. High Angle, Join one node directly, brace across prior bay(s)



D. Terminate at points, single node joints

Figure 9. Truss-to-truss joint geometry candidates.

After our initial investigation, we performed additional modeling using sections of truss that were fabricated using the Trusselator prototype constructed in our Phase I SBIR effort. Some of these assemblies were single intersections (connection of 2 discreet trusses), and others were multiple intersections (connection of 3 or more discreet trusses in the same area). Each of the assemblies included both butt joint attachments and spanner joint attachments. These models, shown Figure 10, further illustrate the complex geometry possibilities, and the challenge of designing and building a SpiderFab tool which can accomplish this task.

These geometries illustrate the butt joint and spanner joint methods of attachment using CF/PEEK rod or tape segments for the attachment material. Where two nodes have physical contact there would be a single segment that bonds the two together, whereas for the spanner joints, two or three discreet segments would proceed to the nearest nodes, forming a triangulated structure. A "longeron lamination" method, depicted in Figure 10, is used to affix the connecting member onto the longeron of the pre-existing truss structures.

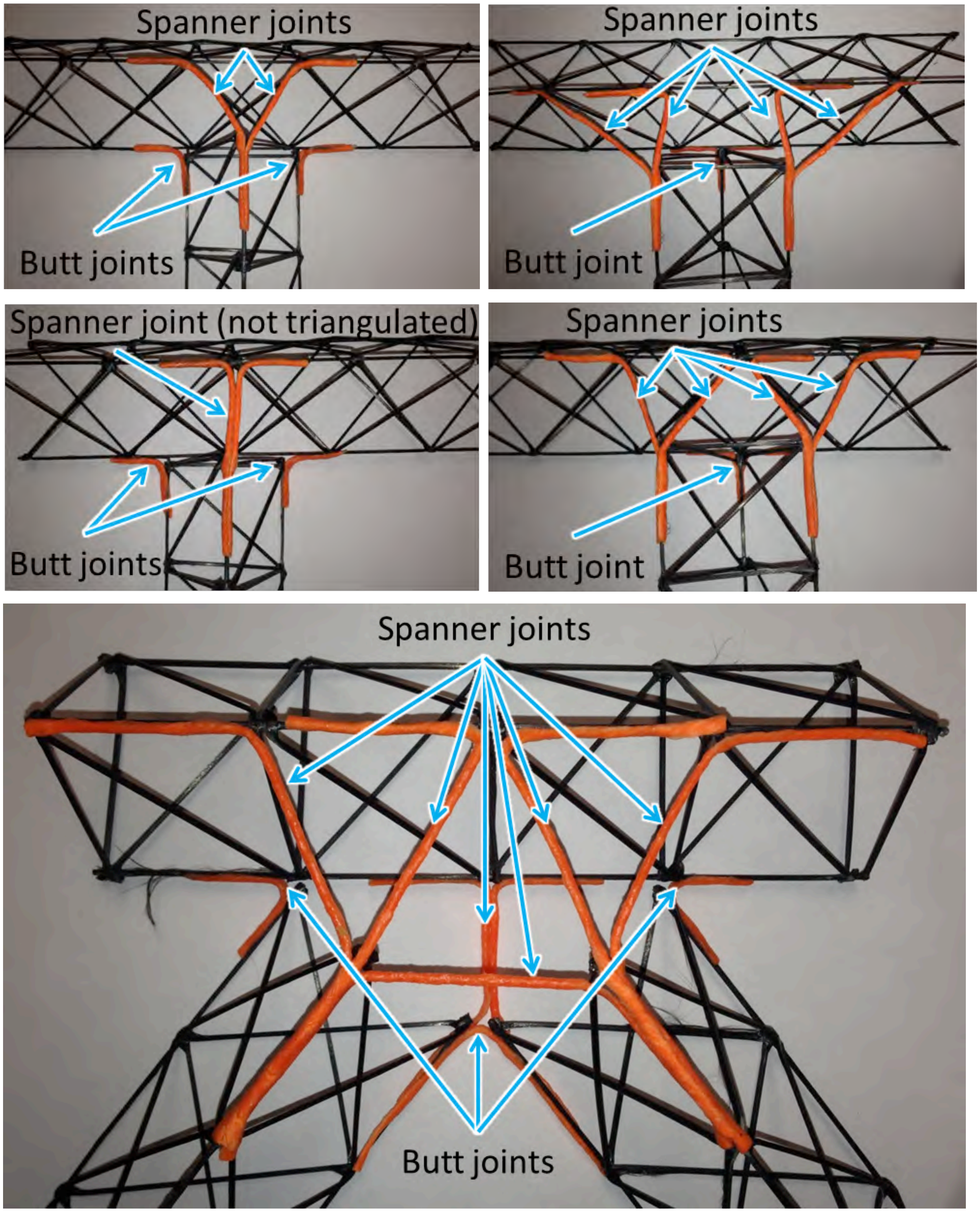


Figure 10. Additional Joint Geometry Concepts. *Models of single intersections (top) and multiple intersections (bottom) using CF/PEEK trusses and flexible rods representing CF/PEEK joining members.*

2.1.1.2 Joint Geometry Fabrication Tests (Butt Joints)

Next we began testing our geometries using CF/PEEK rods and tapes to join pre-made trusses. In the process we tested several of our previously identified critical variables, including feed-stock format (tape and rod), heating method (contact heating), and compaction method (stationary, rolling, and sliding compression). For creating a butt joint assembly, we placed CF/PEEK ribbons across the nodes to be joined, and then pressed a heated metallic block onto them to melt and fuse the PEEK resin. Because the joining material was in ribbon form, these joints relied upon the physical contact of the trusses for compressive stiffness. Figure 11 shows two joints accomplished by this method.

One of the main challenges with this approach was getting the ribbons to adhere to the truss rather than the heated iron. Because of this issue, our compression method transitioned to something more akin to sliding or laminating with a brush-like stroke along the length of the ribbon. This achieved fairly strong bonds, but soon led to build-up of matrix material on the hot iron. Tapes were also added to bridge between bays farther away from the butt joint to act as tension members for additional stiffness. This had a moderate benefit, but it was clear that using pure tension members had limitations. It is clear that with the current truss design, we cannot rely on butt joints alone, as it puts severe restrictions on the location and angle at which the trusses can be joined, which is an unacceptable limitation for long-term goal of the SpiderFab process to be capable of fabricating structures with geometry varied throughout their extent in order to optimize their performance.

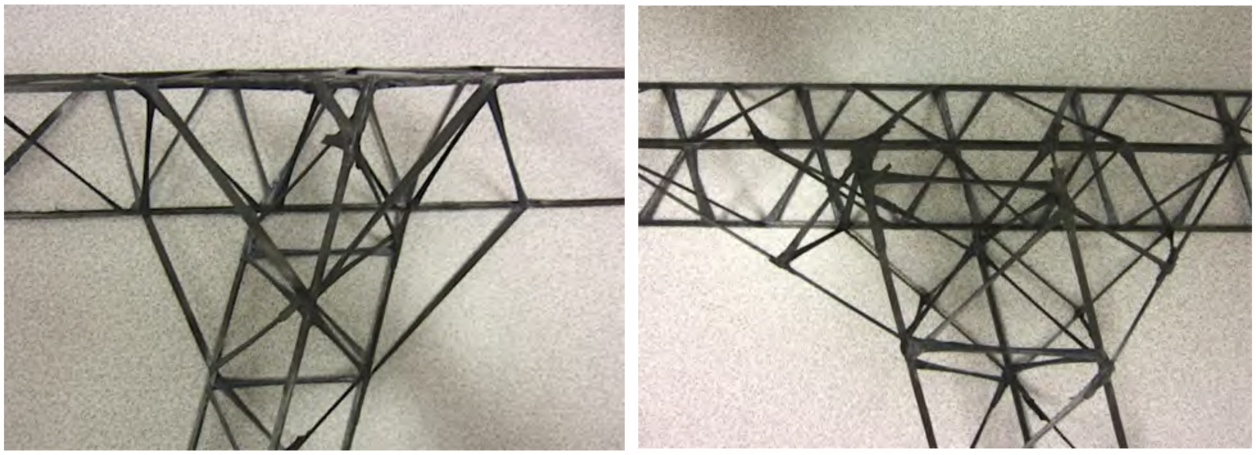


Figure 11. Butt Joint Geometry. *Joining was done using a heated iron to attach CF/PEEK ribbons (left) and rods (right) from the nodes of one truss to the nearest nodes of the other. Note the angle of attachment determined by the location of the nodes, illustrating a sever limitation of using only butt joints.*

2.1.1.3 Joint Geometry Fabrication Tests (Spanner Joints)

To create an assembly in which the two trusses can be joined at any angle or orientation, they must be separated by a gap and a spanner joint method employed. This necessitates members with compressive strength, so CF/PEEK rods were used rather than tapes. Figure 12 shows a top and bottom view of an assembly of two trusses utilizing the spanner joint method.

The same hot iron was used for heating and joining the spanning segments. Using CF/PEEK in the rod form had the advantage of not adhering to the hot iron as much as the ribbons, but after the initial fusing of the joint, some "brush stroke" motion was helpful for getting all the fibers to lay down neatly and forming a strong bond. This spanner joining method showed considerable improvement over the previously tested butt joint method with ribbon material.

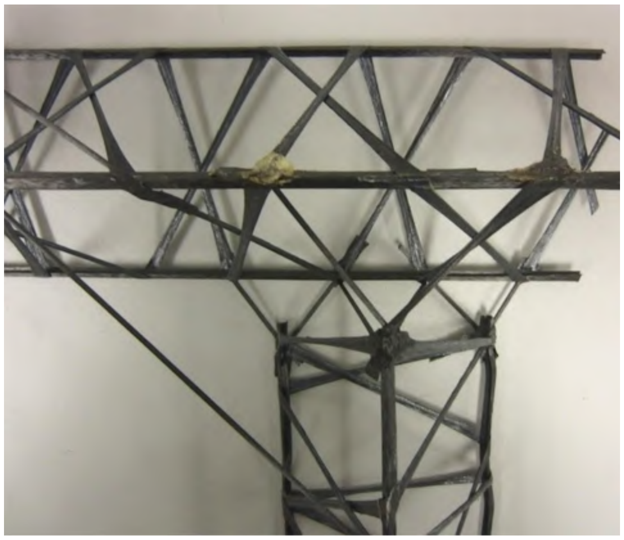


Figure 12. Spanner Joint Geometry. *Joining trusses with a separation distance was accomplished using a heated iron to attach CF/PEEK rods from the nodes of one truss to the nodes of the other (spanner joint).*

2.1.1.4 Joint Geometry Fabrication Tests (Splice Joints)

After experimenting with making assemblies of trusses, we experimented with several other applications of the contact welding method using the hot iron. Two important uses are for the repair of broken joints within a truss, and for splicing of segments together. To test the capability of the contact welding approach for these needs, we cut one longeron of a truss in multiple places along its length. Using heat and compression we were able make lap joints to repair the longeron in a shortened state, resulting in a curved truss, shown in Figure 13. This could be a useful method for creating structures with organic shapes or complex geometries.

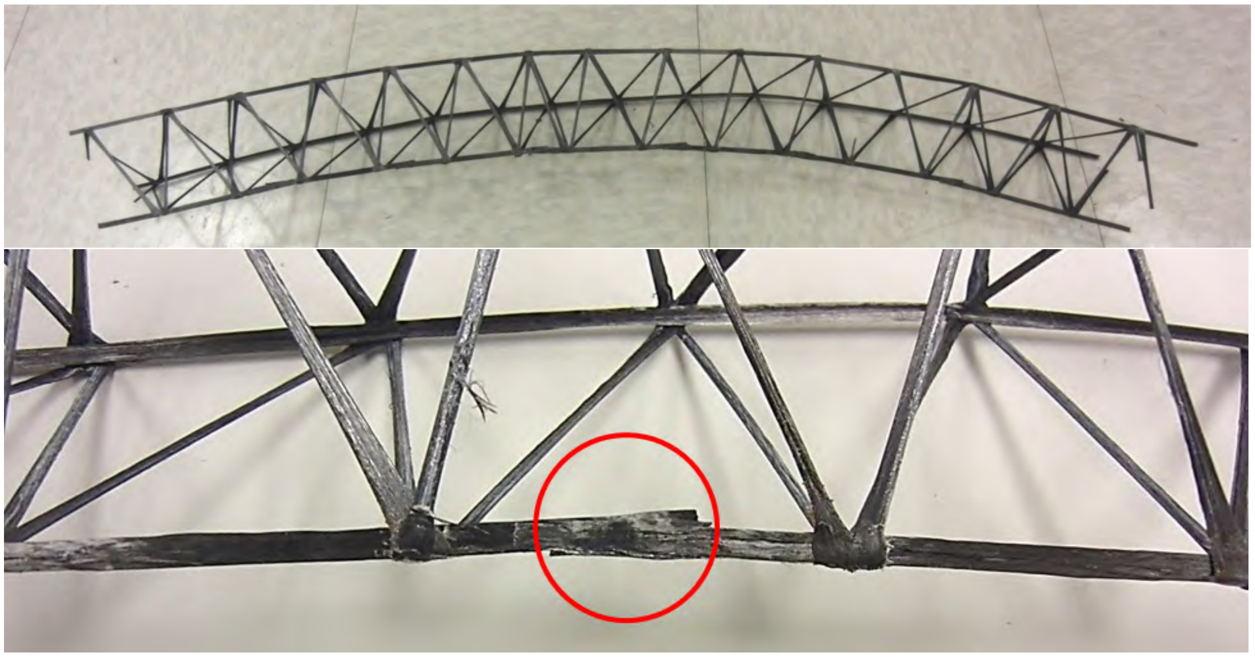


Figure 13. Curved truss-making concept. *Curved truss formed by cutting every other bay of one of the longerons, and rejoining it in a shortened state (circled). Rejoining was done using a heated iron.*

2.1.1.5 Joint Geometry Fabrication Tests (Point Joints)

After achieving encouraging results with the previous tests, we evaluated the capability of our process to form tapered ends onto pre-existing trusses. Tapering the ends of the trusses would enable truss elements to be connected with point joints, which have the advantage of transmitting only tension and compression, thereby simplifying structural analysis. On one end of the truss we clipped off the diagonals and bent the now unsupported longerons radially inward until they met in a point. Bending the longerons required heating at the last remaining node to soften the material. The three were joined in a point by heated contact using the soldering iron in similar fashion to the previously performed tests. Pre-formed CF/PEEK rods were attached between the tapered longerons to provide additional bracing of the structure. Figure 14 (left) shows this tapered longeron point design.

On the other end of the truss we used three longeron sections that had been cut out of another truss, attached one end of each to the final truss node, and joined the other ends to each other at a point on the neutral axis of the truss. This section was also buttressed by additional rods. We also experimented with adding PEEK resin via a manual fused filament fabrication technique, as reinforcement for the joints. Figure 14 (center) shows the resulting termination.

Finally, this truss was placed in TUI's Instron Machine for compressive strength testing, shown on the right of Figure 14. Each time the truss reached its limit and broke, we repaired and reinforced it for further testing, resulting in 18 successive compression tests, with a maximum load measurement of 208 lbs. The capability of the SpiderFab methods to repair and reinforce weak or broken segments will be a major factor for risk reduction for future flight opportunities.

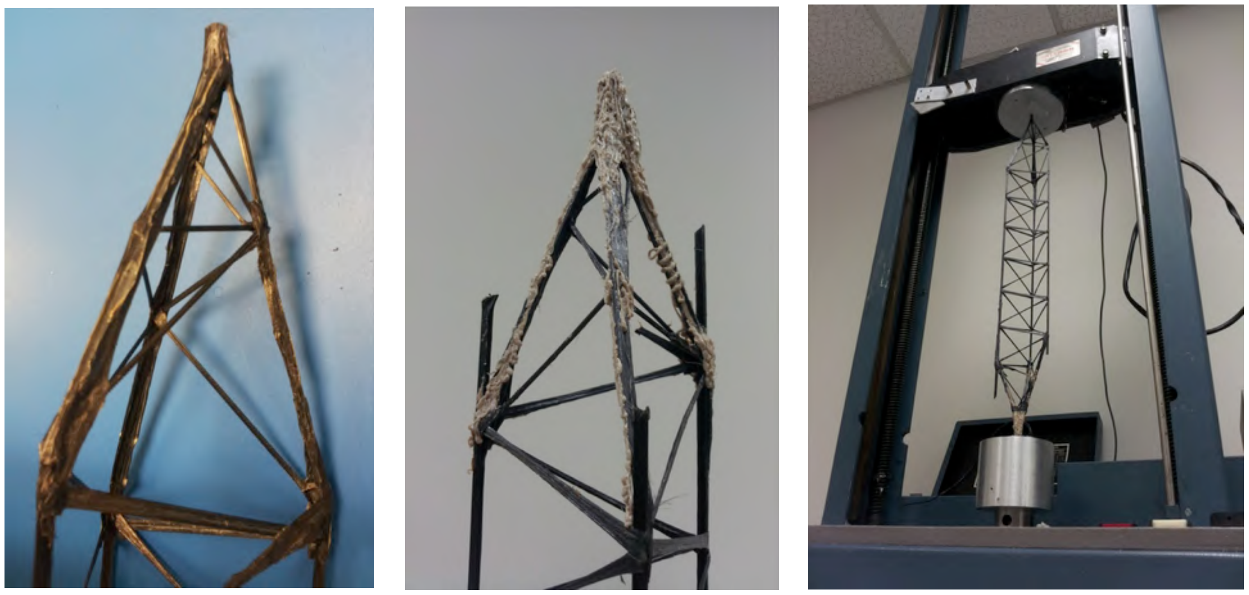


Figure 14. Point Joint Geometry Fabrication and Testing. *Using SpiderFab techniques, we formed pointed tips on the ends of a truss (left), reinforced them with PEEK resin (center), and performed compression to evaluate their structural integrity (right).*

2.1.2 SpiderFab Joiner Spinneret Prototype Development

Having observed and evaluated some of the significant variables for successful manual application of the SpiderFab processes, we were ready to begin designing and building hardware to perform these processes using robotic automation. We started by sketching out several design concepts for a SpiderFab tool which would do one or more of these things: feed material into a processing zone, form tape into rods, hold the material against the application surface, heat it to above the melting temperature of PEEK (380C), cool it to below the service temperature of PEEK (250C) under compaction, cut the feedstock, and restart another segment of feedstock.

One of the problems that we hoped to solve in the development of a SpiderFab Joiner Spinneret was the adhesion of matrix material to any surface other than the intended truss joint. This would include the heated contact welder as well as any heated forming elements such as those for transitioning tape into rods. An idea was generated that we could perform both heating and cooling with the same compression block if we could dump heat quickly with active cooling. This method is fairly inefficient from a total power perspective, but it has the potential to solve one of the major issues that we had previously encountered.

We quickly sketched out a simple concept and made a prototype to test, as shown in Figure 15, which included a heated block that would melt the PEEK, and an actively cooled plate that would be brought to bear on the heated block for quick cooling and solidification of the matrix, and a conical spring to separate the two except under sufficient compression. Along with a strategic heating and cooling routine, this has enabled removal of the contact surface without adhesion of material.

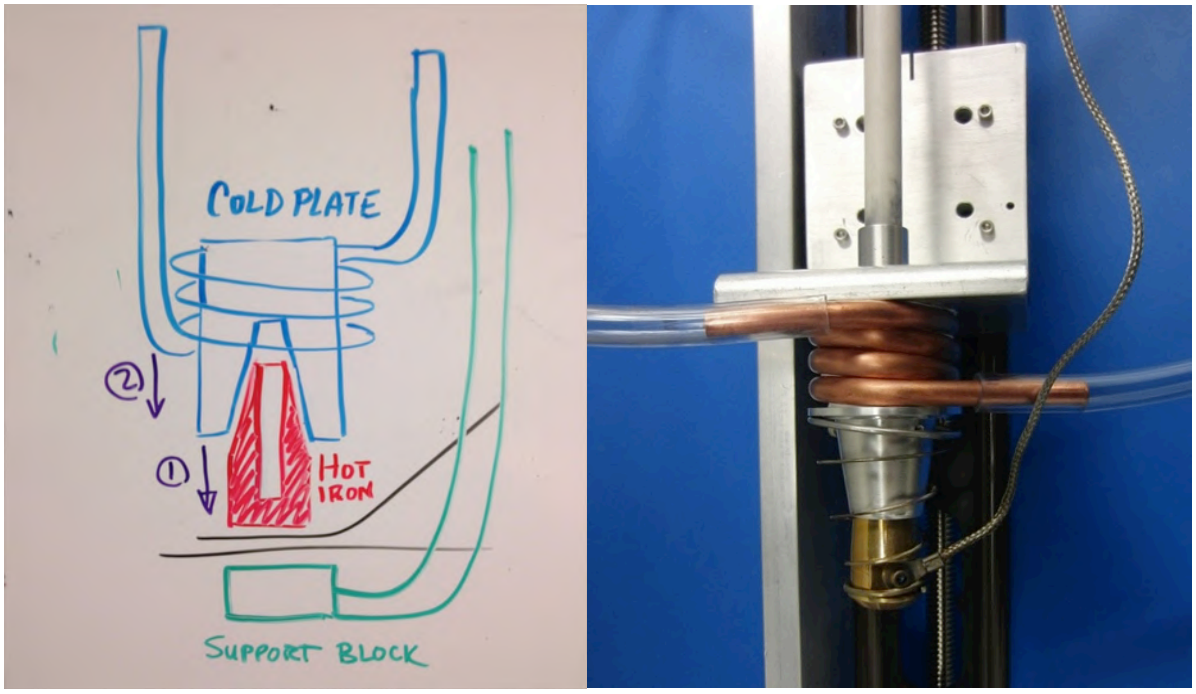
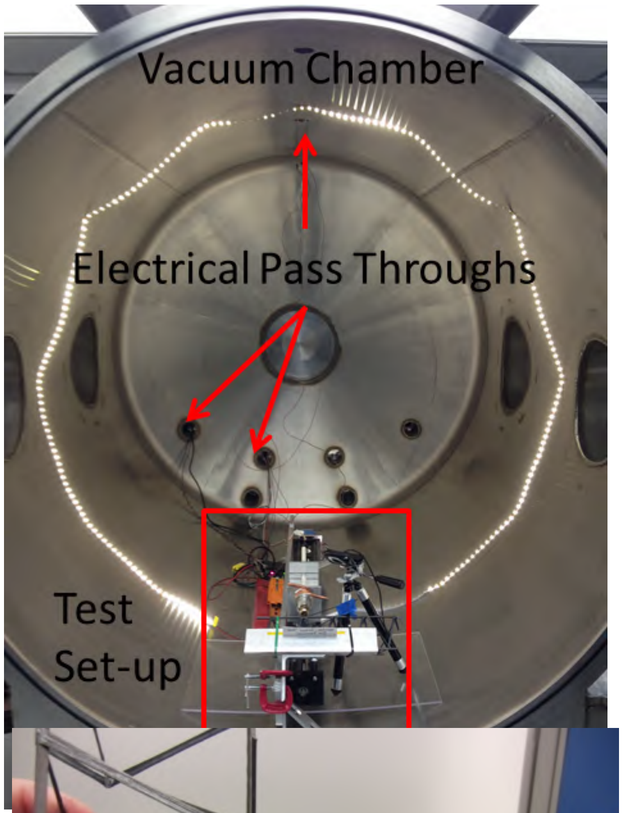


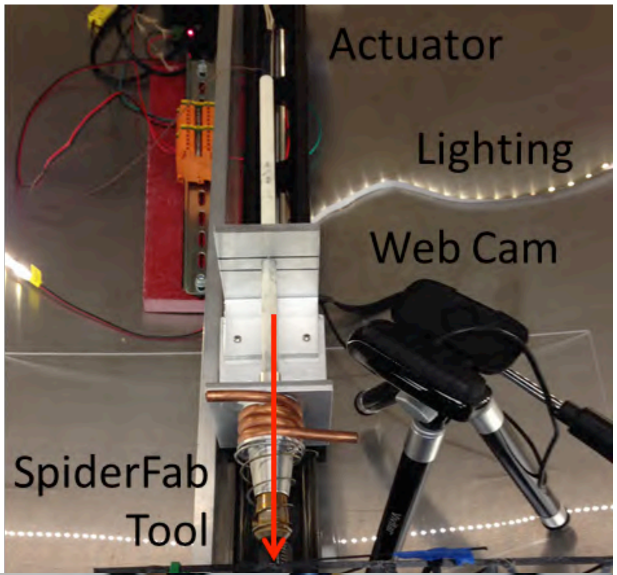
Figure 15. Hot / Cold Welder Concept (left) and Prototype (right). *This contact welder heats the material to the melting point, shuts off the heater, applies the cold block to draw away heat, and retracts from the joint with little to no adhesion of matrix on the welder head.*

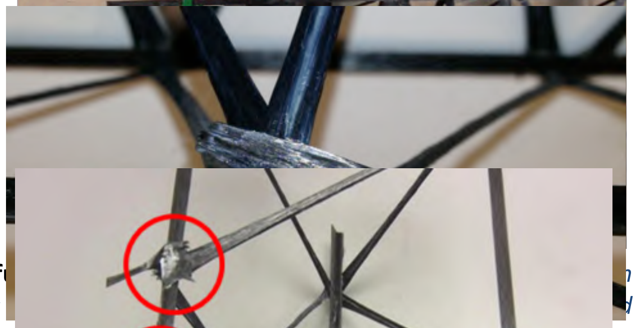
Error! Reference source not found. shows preliminary testing (left) and the resultant joints formed with consolidated rod stock (right). The resulting joints seem to have excellent consolidation, and more-than-sufficient strength. The rod flattens out with heated compression, resulting in a larger surface area for good cohesion with the truss.

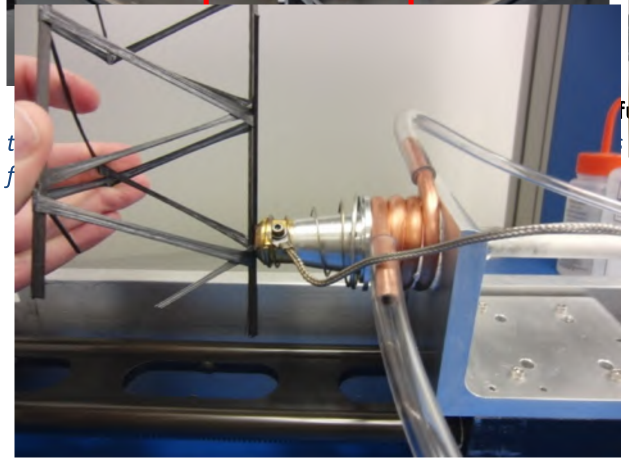
2.1.3 SpiderFab Joiner Spinneret Prototype Vacuum Testing

The objective of this test was to investigate the ability of the SpiderFab Joiner Spinneret to rapidly form a weld between CF/PEEK materials and then release the tool from the composite material without accumulating PEEK on the tool, all while under vacuum. For expediency, these tests were not performed with water-cooling to eliminate the need to make water pass-throughs in and out of the chamber. For automation, the tool was mounted to a lead screw actuator controlled via laptop from outside the chamber. The actuator forced the welding head against a fixtured CF/PEEK truss, with a rod of CF/PEEK situated in between. A test stand was setup in the vacuum chamber with integrated lighting and video camera, as shown in Figure 17.









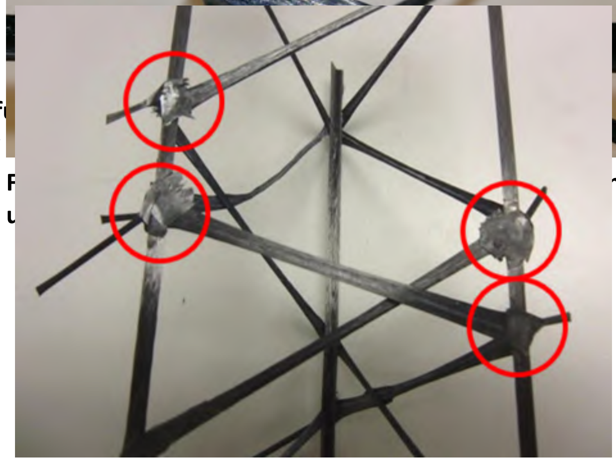


Figure 16. Joiner Spinneret (left) welding CF/PEEK rods to the nodes of a truss (right).

Several test welds were made at atmosphere to verify operation of the tool and to calibrate the actuator. After successfully proving the system in atmosphere, the same test was performed in vacuum of 300 mtorr. The entire welding process at atmosphere with no water cooling was approximately 2 minutes, while the time in vacuum was approximately 5 minutes, due to the lack of convective air cooling. The addition of liquid cooling for vacuum testing will significantly reduce cycle time. The joints formed in vacuum were visually of equal quality to those formed in atmosphere, achieving good consolidation with the truss, as shown in Figure 18.

2.1.4 SpiderFab Joiner Spinneret Engineering Model

After the success of the first generation prototype, we decided to build a second Joiner Spinneret, integrating improvements based upon our results testing the first prototype. The new tool operates in the same manner as the previous one, but with some improvements for performance and automation. For one, the weld head mass has been reduced from 80g to 30g, which allows it to thermally cycle much more quickly and power-efficiently. The new design has a small linear stage and 12V linear actuator for driving the motion of the weld head against the substrate. Lastly, the new design has a prong which extends from the base of the tool out in front of the weld head to provide a support structure against which the mechanism can push to compact the joint. This prevents any sagging or collapsing of the truss due to imbalanced forces while it is in a softened state. The new design can be attached to a pistol grip handle for hand-held operation, or to a mounting plate for a robotic arm. Figure 19 shows a cross-section view of the new design.

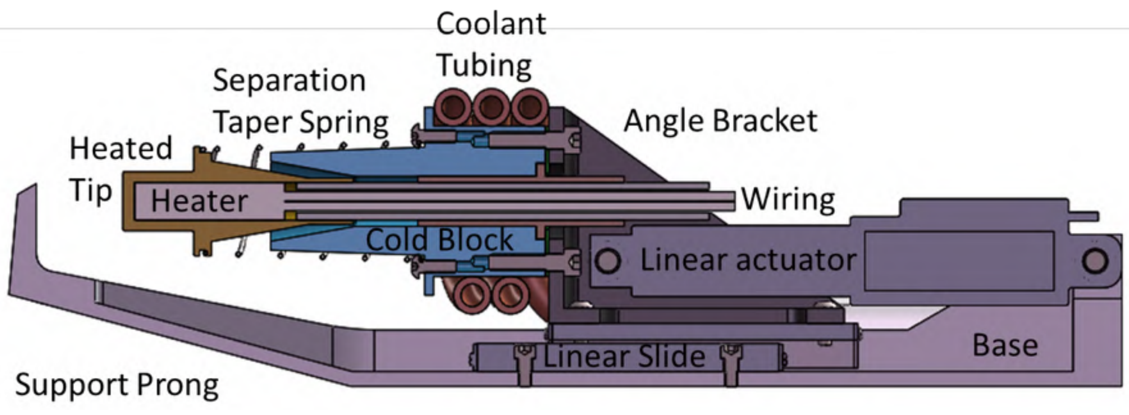


Figure 19. Cross-Section of the new Joiner Spinneret with Linear Actuator and Support Prong.

This version is electronically controlled by an Arduino microcontroller, which is programmed to govern the staged operation of the system:

1) Fully retract the linear actuator
2) Turn on heater (temperature controller set to 400^{o}C, measured by a thermocouple)
3) Extend the linear actuator until the tip contacts the elements to be joined
4) Hold for 5 seconds to melt the PEEK resin
5) Turn off the heater
6) Extend the linear actuator until the cold block seats against the back of the heated tip
7) Cool tip for solidification of PEEK
8) Loop back to step 1

When the tip is released from the composite joint, the separation spring returns the heated tip to its fully retracted position, and the tool is ready to perform more joints.

The Arduino interfaces with devices for heating, temperature sensing, linear actuation, and display: 1) a solid state relay turns the cartridge heater on and off during various phases of heating and cooling, 2) a thermocouple amplifier breakout board converts the voltage from the thermocouple to a resolution that the Arduino can read, 3) a control board governs the motion of the linear actuator at the direction of the Arduino, and 4) an LCD screen displays the status of the Joiner tool. Power comes from a standard wall outlet through a 9 Volt DC adapter.

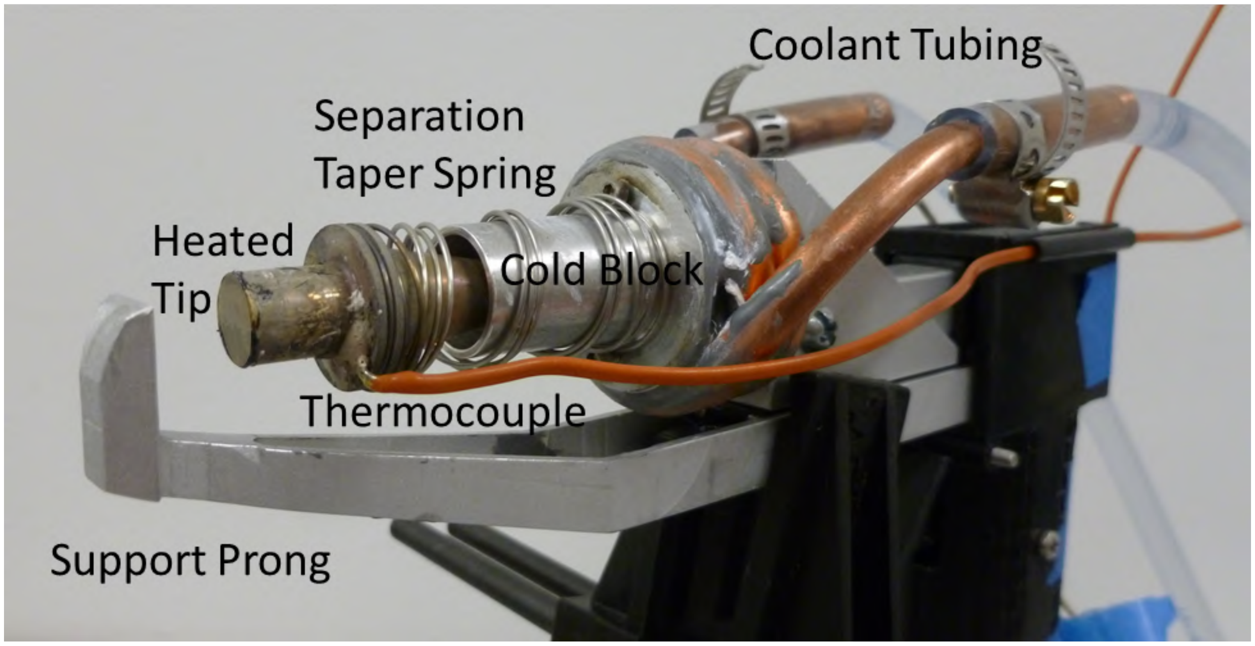


Figure 20. SpiderFab Joiner Spinneret. *TUI developed a custom contact welder which can heat and cool the joined substrates under compression, resulting in strong bonds and minimal weld head adhesion.*

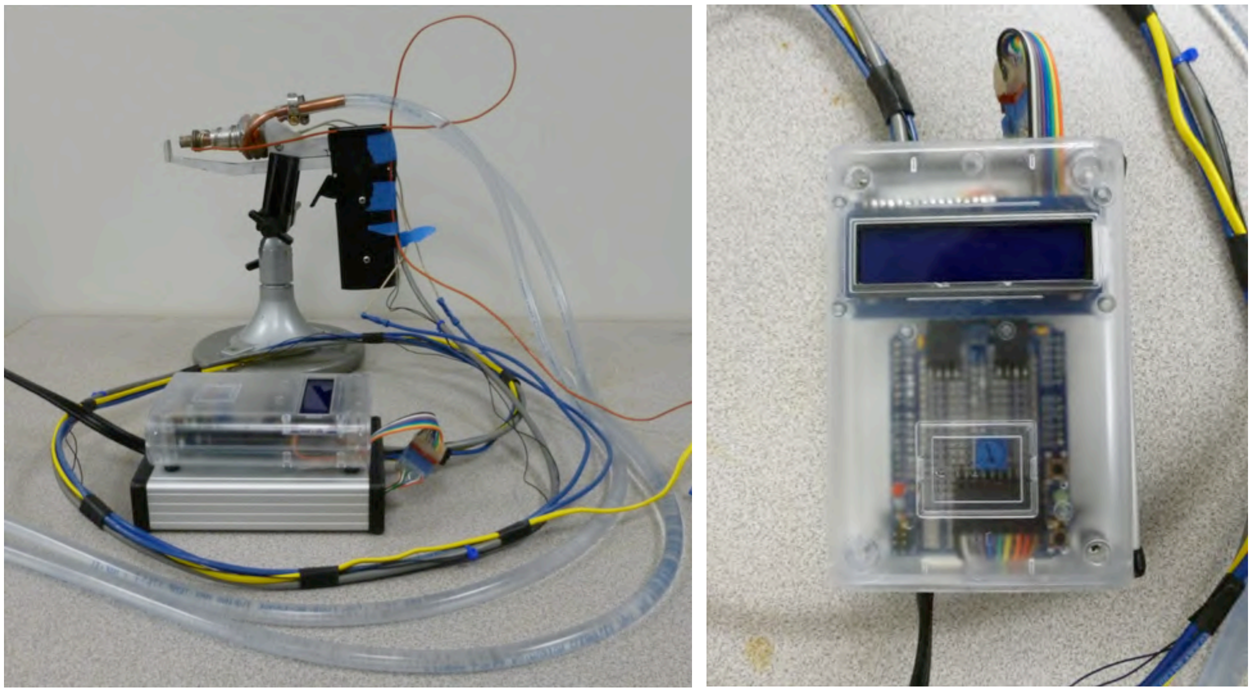


Figure 21. SpiderFab Joiner Spinneret Tool with Water Cooling Lines and Control Electronics. *The water lines draw heat away from the contact welder for re-solidification of the PEEK matrix. The Joiner Spinneret has three phases of operation, governed by an Arduino microcontroller.*

2.1.5 SpiderFab Pultrusion Spinneret Tool

We also prototyped a pultrusion tool to form tension members and extrude free-form rigid members. The 3D-printed prototype pultrusion 'spinneret' end effector is shown in Figure 22. This tool forces co-mingled glass fibers and ABS through a heated die to form a consolidated extrusion with round cross-section. Figure 23 shows a 'fractal pyramid' structure fabricated using this tool, and Figure 24 shows a test in which this tool was used as an end-effector on the Baxter robot.

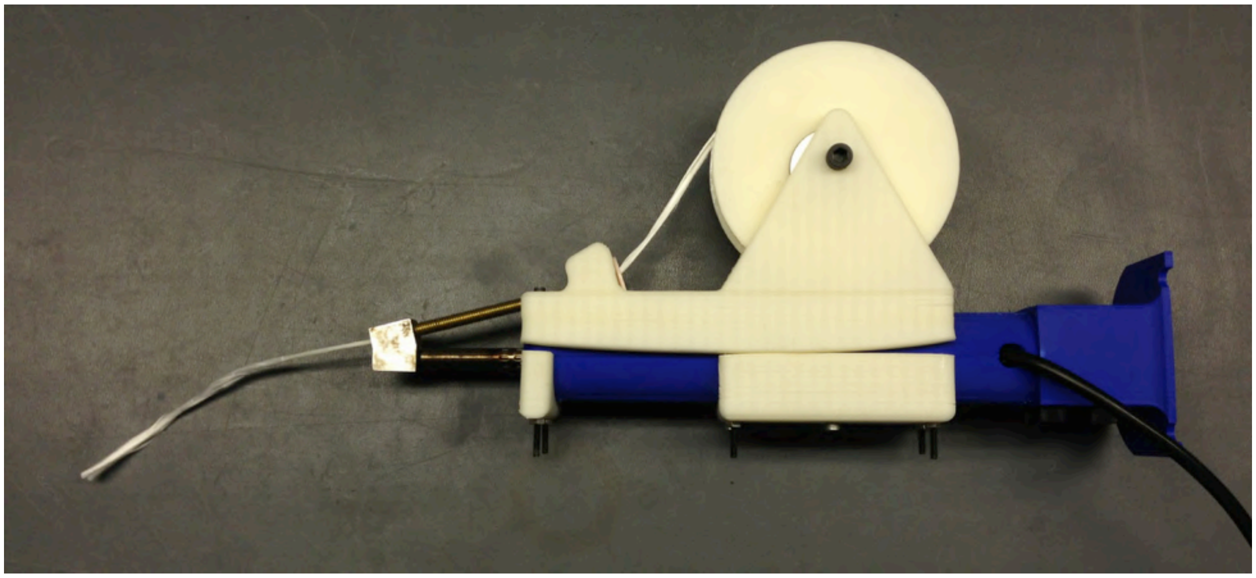


Figure 22. SpiderFab Pultrusion Spinneret Tool. *A spool of flexible co-mingled glass and plastic fibers get pulled through a heated die for melting and consolidation into stiff members for tension and moderate compression.*

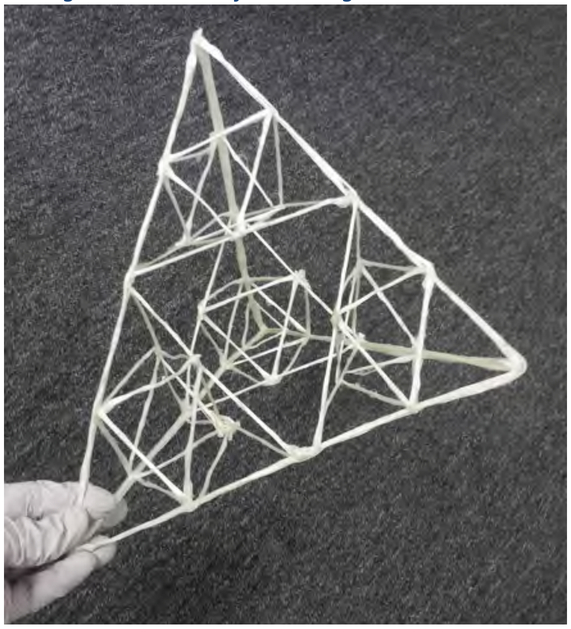


Figure 23. A 'fractal pyramid' constructed of Twin-tex material using the SpiderFab pultrusion spinneret.

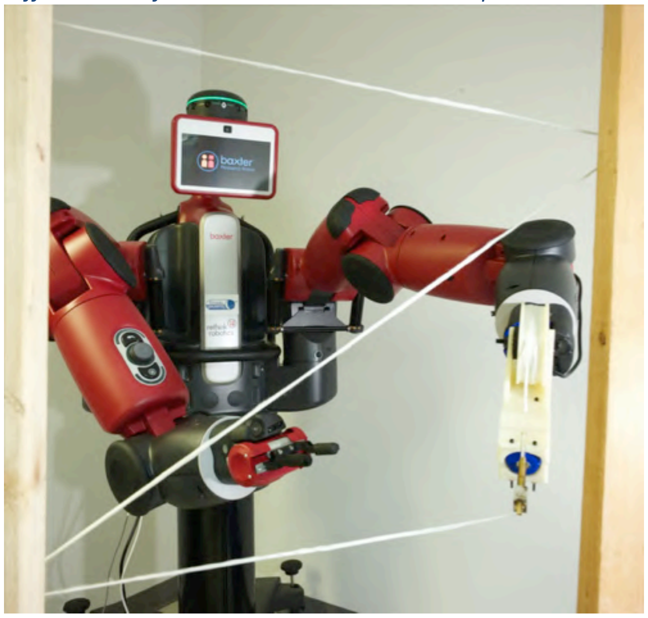


Figure 24. Test of pultrusion of long elements using the SpiderFab spinneret as an end-effector on the Baxter robot.

2.1.6 2nd Order Truss Assembly Demonstration

In order to scope the challenges that must be addressed to enable robotic assembly of 2nd order truss structures, we fabricated multiple segments of truss with our Trusselator prototype and then manually assembled a truss-of-trusses structure using an assembly jig constructed of 80:20 components, shown in Figure 25 and Figure 26, the Joiner Spinneret prototype shown in Figure 20, and the Pultrusion Spinneret shown in Figure 22. The purpose of this exercise was to characterize the dexterity, reach, and range that a robotic manipulator will require to enable such an assembly. Figure 27 shows the 3-m long 2nd-order truss sample, which masses just 620 grams.

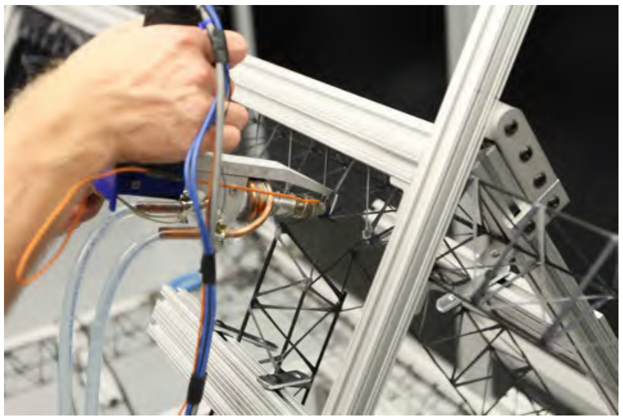


Figure 25. 2nd-Order Truss Assembly. *The Joiner Spinneret was used to join longeron and batten 1st order truss elements, and a Pultrusion Spinneret used to create diagonal tension elements.*

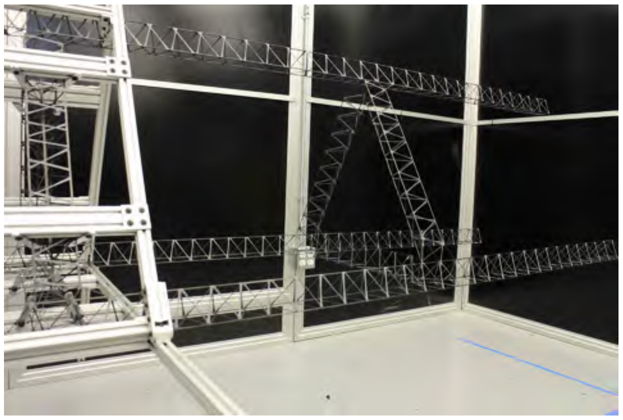


Figure 26. A jig was used to position the elements for welding.

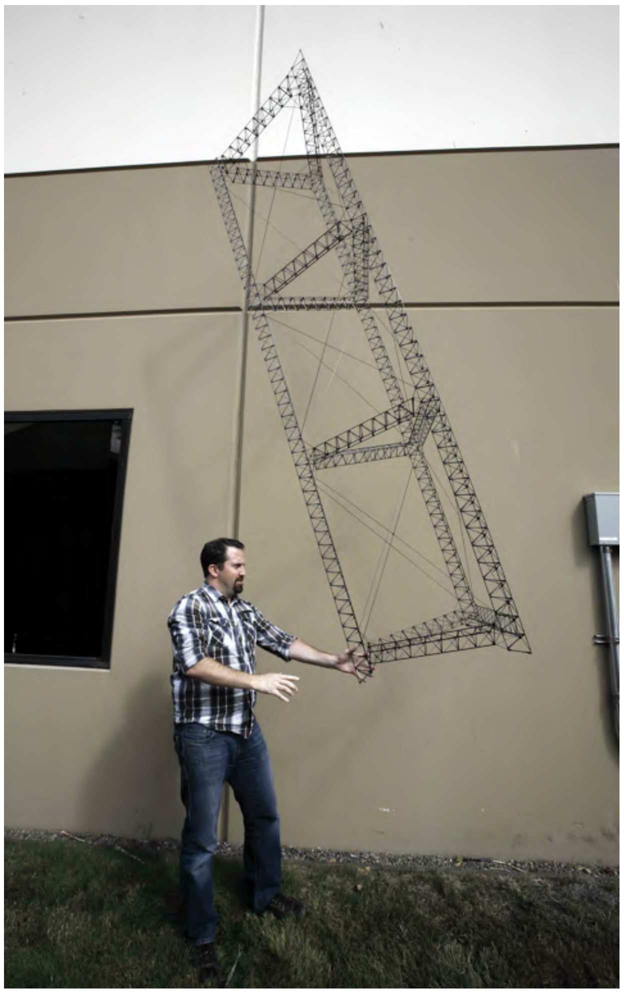


Figure 27. Truss-of-trusses sample. *The 2nd order truss sample is 3m long and masses just 620 grams.*

2.2 Development of Robotics Methods for Assembly of Space Structures

Having demonstrated the feasibility of assembling composite truss based structures using relatively simple tools employing thermal processes, we turned our focus to developing methods to enable robotic systems to perform assembly of these structures in a highly automated manner.

2.2.1 Overview

The focus of this effort was on performing proof of concept demonstrations that verified the ability for an autonomous robot to grasp, manipulate, and join trusses. To complete these demonstrations, we developed a fast and efficient robotic vision system to enable closed-loop control of the robotic assembly, developed a robust software framework to provide support for these and future robotic demonstrations, designed, fabricated, and tested custom robot end-effectors and truss joints.

2.2.2 Baxter Robot

To support these demonstrations, TUI acquired, under company investment funds, a Baxter robot from Rethink Robotics. The Baxter, shown in Figure 28, is a robotic platform combining two 7DOF, 1.2m reach robotic arms and a vision system that includes both head-mounted and arm-mounted boresight cameras. While the Baxter would not be suitable for use in a space environment, it provided a very capable and affordable platform for developing and validating end-effector tools, vision-based software algorithms, and assembly CONOPS.

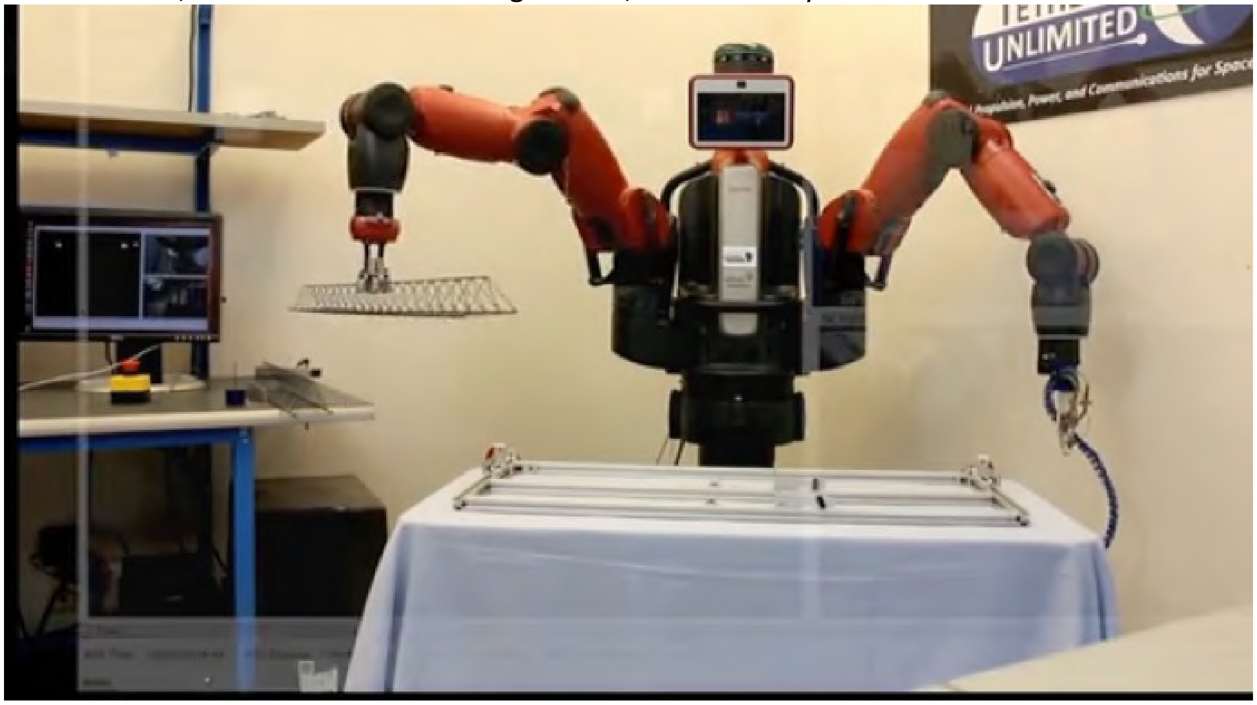


Figure 28. TUI's Baxter robot performing a truss assembly demonstration.

2.2.3 SpiderFab Robotic Vision System

To perform autonomous on-orbit truss assembly and construction, a robotic system will need a sensing system that will enable it to precisely grasp, position, and join truss elements. For this effort, we chose to demonstrate the feasibility of using vision-based software methods to provide this sensing capability. Working with the Baxter robot as a test platform, we developed a custom SpiderFab image processing framework incorporating the following capabilities:

1. Truss detection and location approximation.
2. Reliable closed-loop truss manipulation.
3. Guided truss manipulation.
4. Real-time image processing.

2.2.3.1 Truss Detection

The first step in the development of the robotic vision system was being able to reliably identify a truss in a variety of different environments. Object recognition is a vital component for any machine vision system and recent advances in this field have resulted in a number of popular algorithms and image processing methods being introduced to aid in this task. However, there is no one-size-fits-all algorithm and each object presents a unique and non-trivial challenge when trying to formulate the correct algorithms to identify it within images of near-arbitrary environments.

One method that many contemporary machine vision systems choose to identify objects in non-predetermined environments is by *feature detection* and *feature descriptor matching*. That is, several key visual features of an object (which are ideally unique to that object) are identified using one or more detection algorithms and the resulting feature descriptors are then used to determine if the object is likely present within an image. Color, shape, edges and prominent markings, tags or logos are all commonly used features used by feature detection algorithms. Examples of object detection by feature detection and matching algorithms are illustrated in Figure 29.

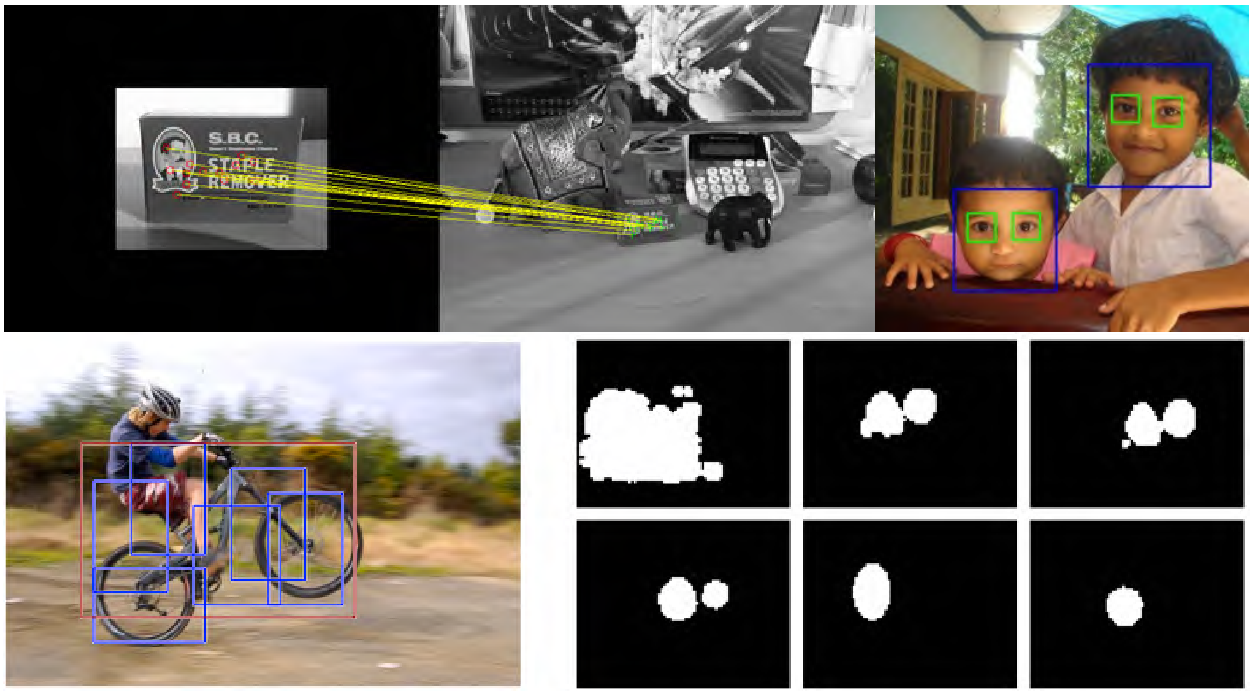


Figure 29. Examples of object detection by Feature Detection and Matching.

Mathematically, there is no universal definition as to what constitutes an image "feature" and calculation time requirements can differ vastly depending on the application. Thus, object detection using feature matching is done in a variety of ways.

To start, we identify the key features of our trusses that can be detected using common image processing techniques. Note that the Python version of OpenCV was used for image processing implementations. Additionally, since the cameras on our current robot platform are limited to monocular vision, algorithms are only performed with respect to 2-dimensional space.

At quick glance, the most prominent features of the truss are the multiple straight-edge longerons that make up the core construction. Out of necessity, the truss segments are arranged in fixed patterns. Currently, they are also uniform in color (dark shade of grey). An example is shown in Figure 30.

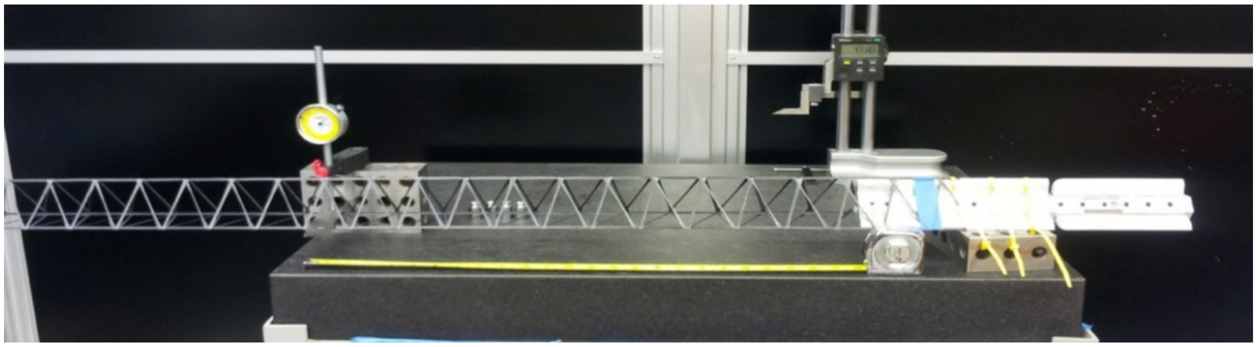


Figure 30. Example truss sample.

Unfortunately, our trusses also have several undesirable properties for many image processing object detection algorithms which would prove challenging to overcome. These include:

- Color – The trusses we used for testing are a very dark shade of grey. To ensure research into this matter was both insightful and realistic, we chose not to change the color our trusses for testing. For image processing applications, this is a notoriously difficult color to work with—particularly in environments with poor lighting or a dark background. Even in properly lit environments, shadows present additional problems and can partially obscure a truss within an image or even be mistaken for the truss itself when using object detection algorithms.
- Lack of 2-D surfaces – Although the truss is unique in shape, the lack of flat, 2-dimensional surfaces means that many feature descriptor matching algorithms may have difficulty identifying a truss unless viewed at the exact same angle, with the same background. That is, the gaps between the truss longerons allows the background environment to appear as part of the truss when viewed in a 2-D image—therefore changes in background environments and even viewing the truss from a slightly different angle can alter the appearance of the truss.

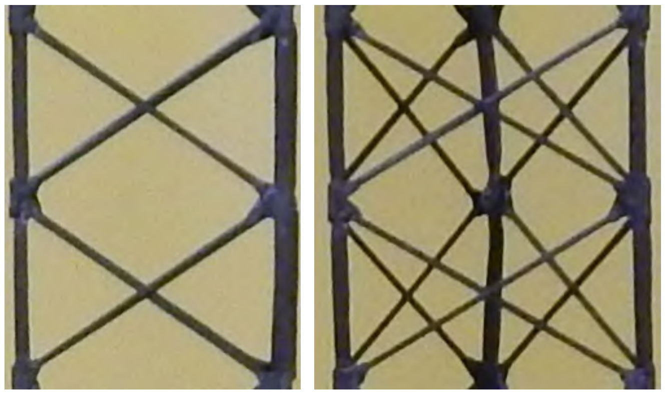


Figure 31. Images of a truss segment viewed from different angles 60 degrees apart.

- Manufacturing imperfections – Despite improvements in the construction and fabrication, trusses will not be identical in appearance.

Despite these disadvantages, significant progress was still made in truss detection using image processing using the algorithms described below.

2.2.3.2 Line Detection Algorithms

The longeron edges can be viewed as individual line segments, which should allow for the use of line detection algorithms. The Hough Line Transform is a simple and popular technique to detect lines within an image. It works on the principle that any line in an image can be represented mathematically as:

$$\rho = x\ cos\theta + y\ sin\theta$$

Where ρ is the perpendicular distance from the origin to the line. The algorithm uses a two-dimensional array, called an accumulator, of (ρ, θ) pairs to determine which pixels (or subset of pixels) in the image are likely to be associated with a straight line. The accumulator forms a sample-space of possible lines (or a random subset in the case of the probabilistic Hough Transform) within an image and each bin in the sample-space with values (or votes) greater than a preset threshold are determined to be lines.

Figure 32 shows the output image of performing a Probabilistic Hough Line transform on an image containing a truss.

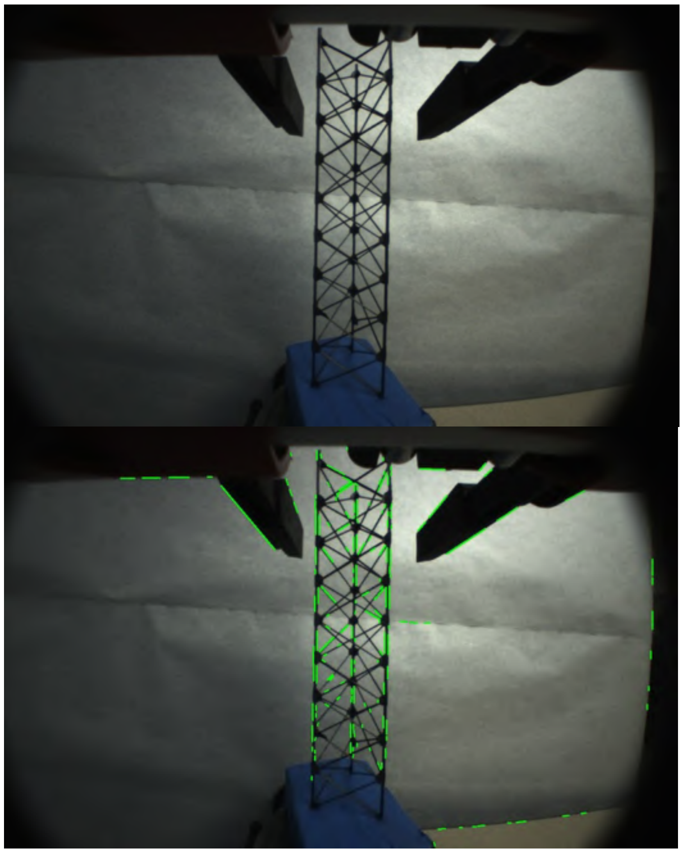


Figure 32. Finding line segments in image using Canny Edge Detection and Hough Line Transform algorithms.

Note that the algorithm successfully detected the edges of the truss, but also picked up considerable noise from lines in the surrounding environment. To filter out the unwanted lines, the inherent properties of the truss must be considered. The three main supporting longerons are paramount to this filtering attempt as they are both the easiest feature of the truss to detect and can also greatly speed up computation time by localizing line segments within a discrete proximity.

To find these longerons, we can perform a histogram of the angles associated with each detected line segment to identify the lines that run parallel with each other. Three (or two) distinct parallel lines of similar length are good indicators of the supporting longerons.

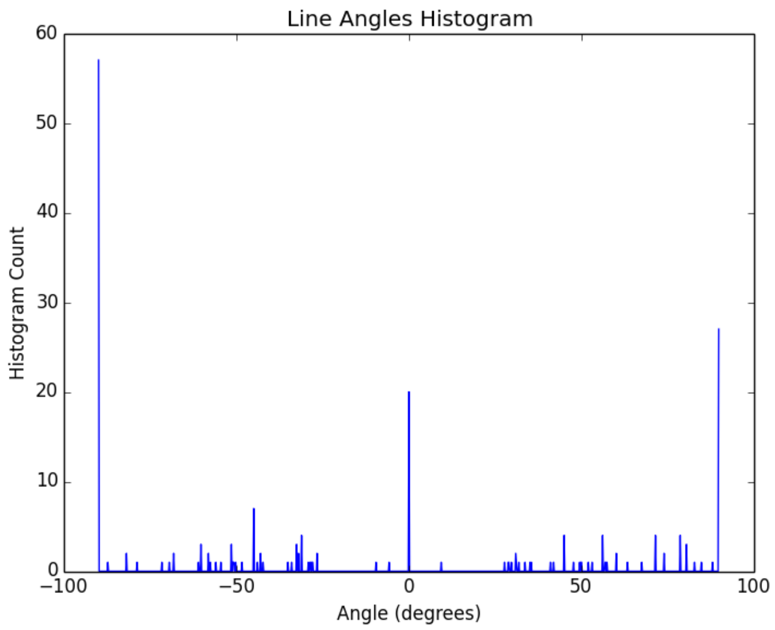


Figure 33 Histogram of Line Segment Angles.

Performing an interpolation algorithm to combine the line segments gives us a better estimate of the longeron lengths and also provides a reduced sample space when attempting to identify the smaller, crossed longerons of the truss.

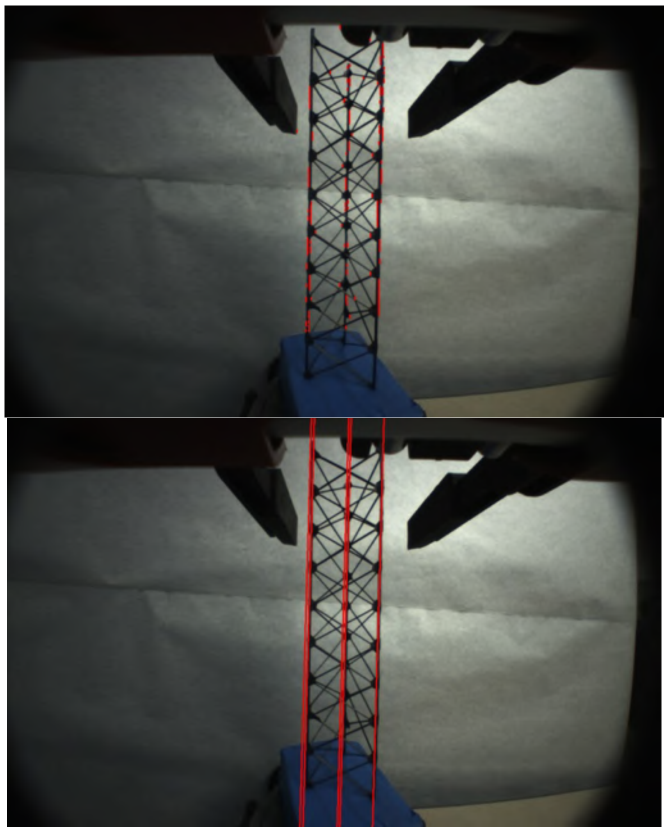


Figure 34. Detecting main longeron supports.

If we examine the region in the image between our newly detected supporting longerons, the smaller longerons of the truss should create a high density of detected line segments. Additionally, these line segments, when taken in parametric equation form, should create several line intersections within our truss region—providing further evidence that what we are looking at is indeed a truss. Note that these line segments should not be parallel with our three main support longerons. Line segments meeting all these requirements are identified as line segments representing a smaller truss longeron.

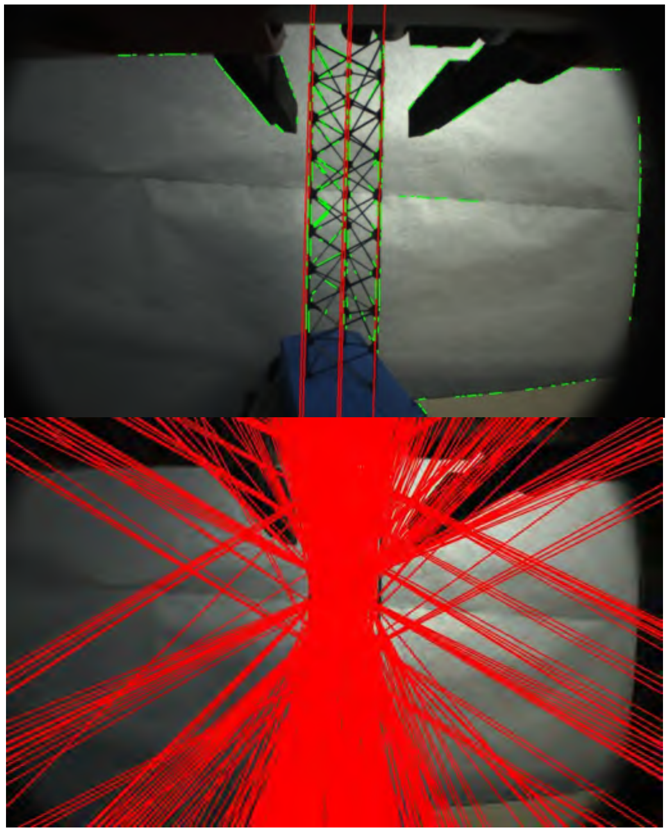


Figure 35. Identifying smaller support longerons. The line segments lying within our truss region and also containing multiple line intersect points within this region are considered part of the smaller support longerons.

Next, we can take all the line segments that we've identified as part of the truss and reconstruct the image using only the truss data points.

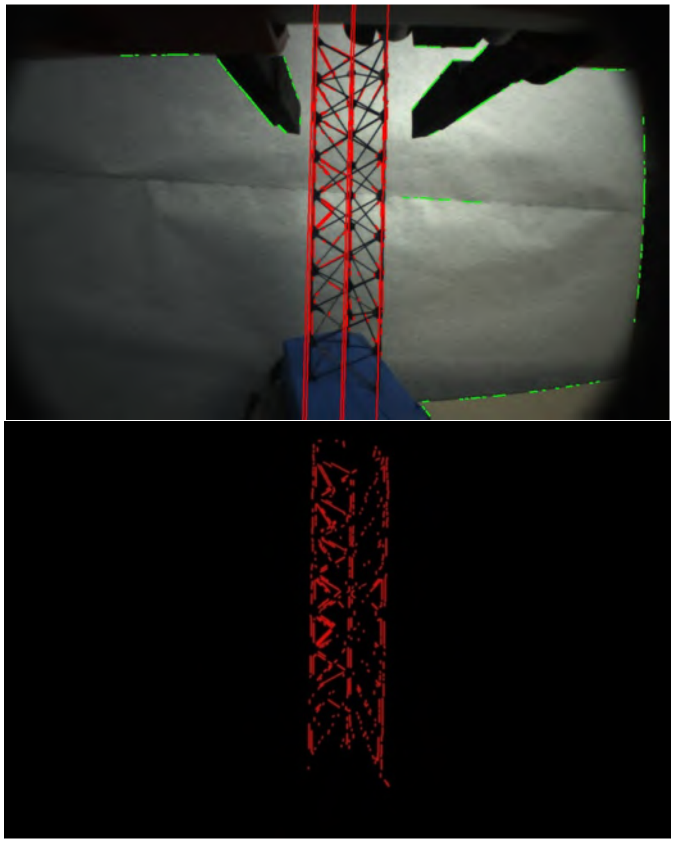


Figure 36. Extracting detected truss line segments from original image.

Finally, applying a Gaussian Blur filter to smooth out the truss line segments gives us our final image.

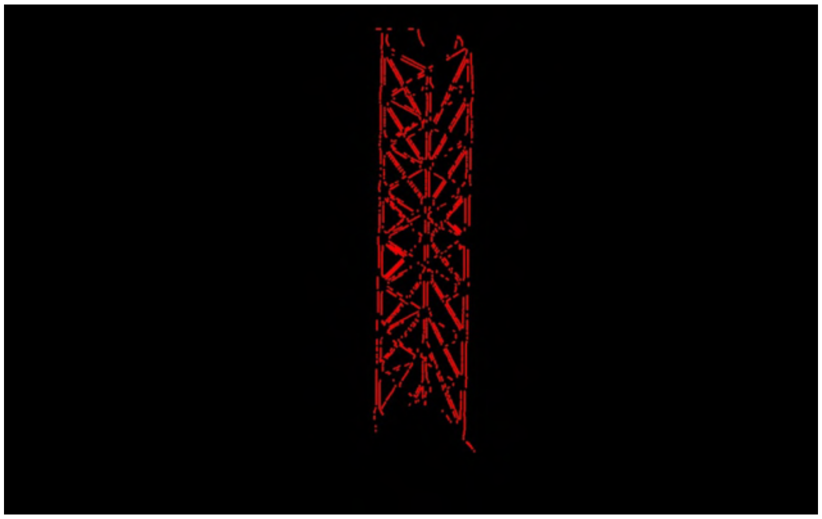


Figure 37. Final image with positive truss identification.

2.2.3.3 Feature Descriptor Matching

Feature descriptor matching is a very popular image processing technique employed by modern machine vision systems. Feature detection algorithms search for prominent or uncommon features within an image using a combination of several advanced image processing techniques.

Information about these detected features is generally stored in *descriptors*—whose size and contents vary according to the feature detection algorithm.

Feature descriptor matching is performed by matching descriptors between two or more images. One image, also called the "query" image, contains the object you are trying to detect. The other image, sometimes called the "train" or "test" image, contains the object within some environment. The feature detection algorithm is run on both images and then a *feature matching* algorithm is used to attempt to match the features shared by the two images—thereby detecting the object in the train image.

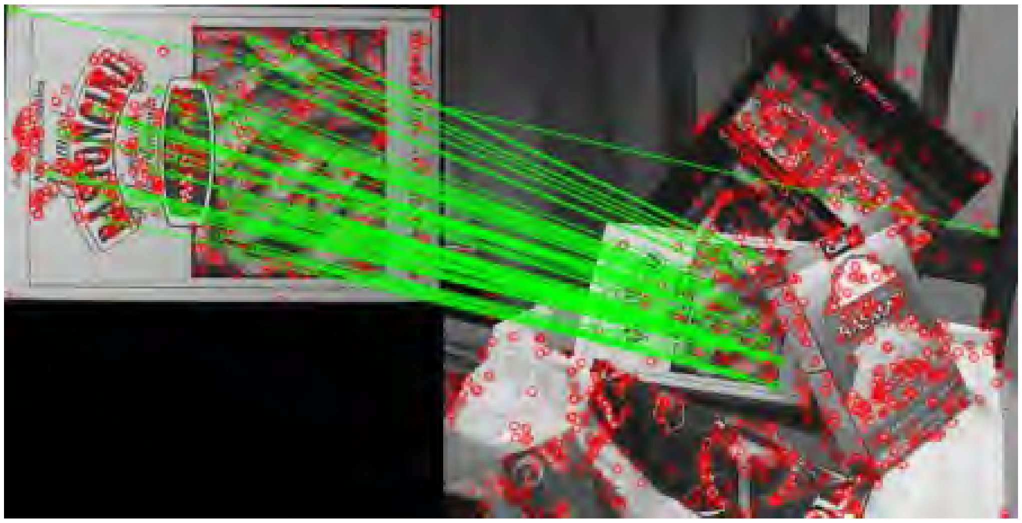


Figure 38. Example of feature descriptor matching.

When choosing feature detection and feature descriptor matching algorithms, it's important that both algorithms perform well given our expected environments and the features of our object. The algorithms should be able to detect our truss in different lighting conditions as well as varying orientations and distances. Ideally, these algorithms should also be fast to meet our real-time image processing requirements.

Three popular feature detection algorithms excel in most of these areas. All three are scale-invariant so they should detect our object at different distances. They are also rotation invariant and consistent at detecting the same features in different illumination and noisy conditions.

1. Scale Invariant Feature Transforms (SIFT) – A very popular algorithm introduced in 1999. This algorithm is very accurate at detecting features in various conditions. Although this algorithm is likely the most accurate of the three examined here, many of the newer algorithms are faster—making them better suited to real-time applications.

2. Speeded Up Robust Features (SURF) – SURF was introduced as a faster alternative to the SIFT algorithm. In an attempt to appeal to real-time applications, it sacrifices accuracy in favor of speed by significantly reducing a features scale-space information by using a convolutional box filter technique.

3. Oriented FAST and Rotated Brief (ORB) – Uses heavily optimized versions of various algorithms with the goal of creating an algorithm faster than SURF and as accurate as

SIFT in most circumstances. It’s important to mention that, unlike SURF and SIFT, ORB is *not a patented algorithm* and can be used without any royalty costs.

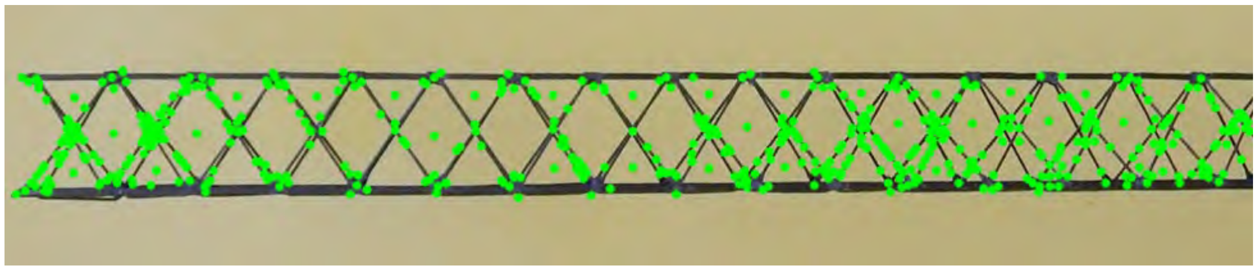


Figure 39. Detecting truss features using SIFT feature detection algorithm. The green circles are features that the algorithm decided are noteworthy and are stored as *keypoint descriptors*.

To perform a more empirical and applicable comparison, each algorithm was used to detect features from a small database of images containing around fifty different truss images. Figure 39 shows an example of the use of the SIFT algorithm on our truss sample. The following graph shows the average computation time of each algorithm when performing feature detection. Note that these tests were performed on a Windows 7 PC with an Intel® Core i7-4770k CPU and 16 GB of memory.

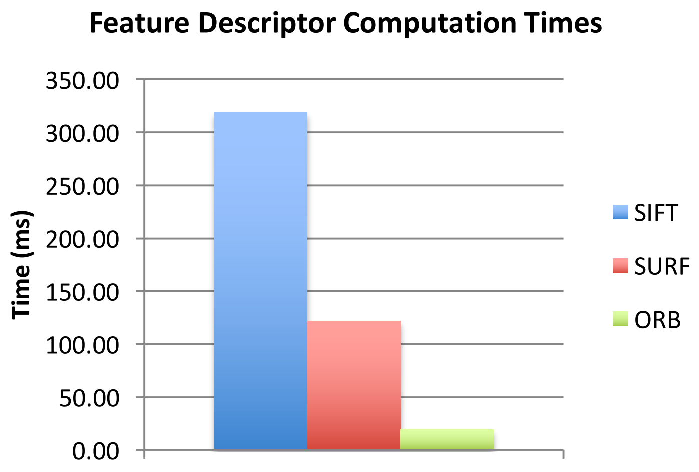


Figure 40. Graph of the computation time of each feature detection algorithm.

As can be seen in Figure 40, ORB was the fastest algorithm by a relatively large margin and SIFT, as expected, was the slowest.

Although speed is an important attribute, we next need to test if the descriptors produced by each algorithm can accurately match objects in different images. Feature matching was tested using two matching algorithms, the Brute-Force Matcher and the FLANN-based matcher. The OpenCV implementations of both algorithms were used for testing purposes.

To continue our comparison testing, we tested different combinations of detection and matching algorithms to attempt to discern which combination would detect our trusses with the most accuracy and which combination would detect them the quickest. Results are compared in Figure 41.

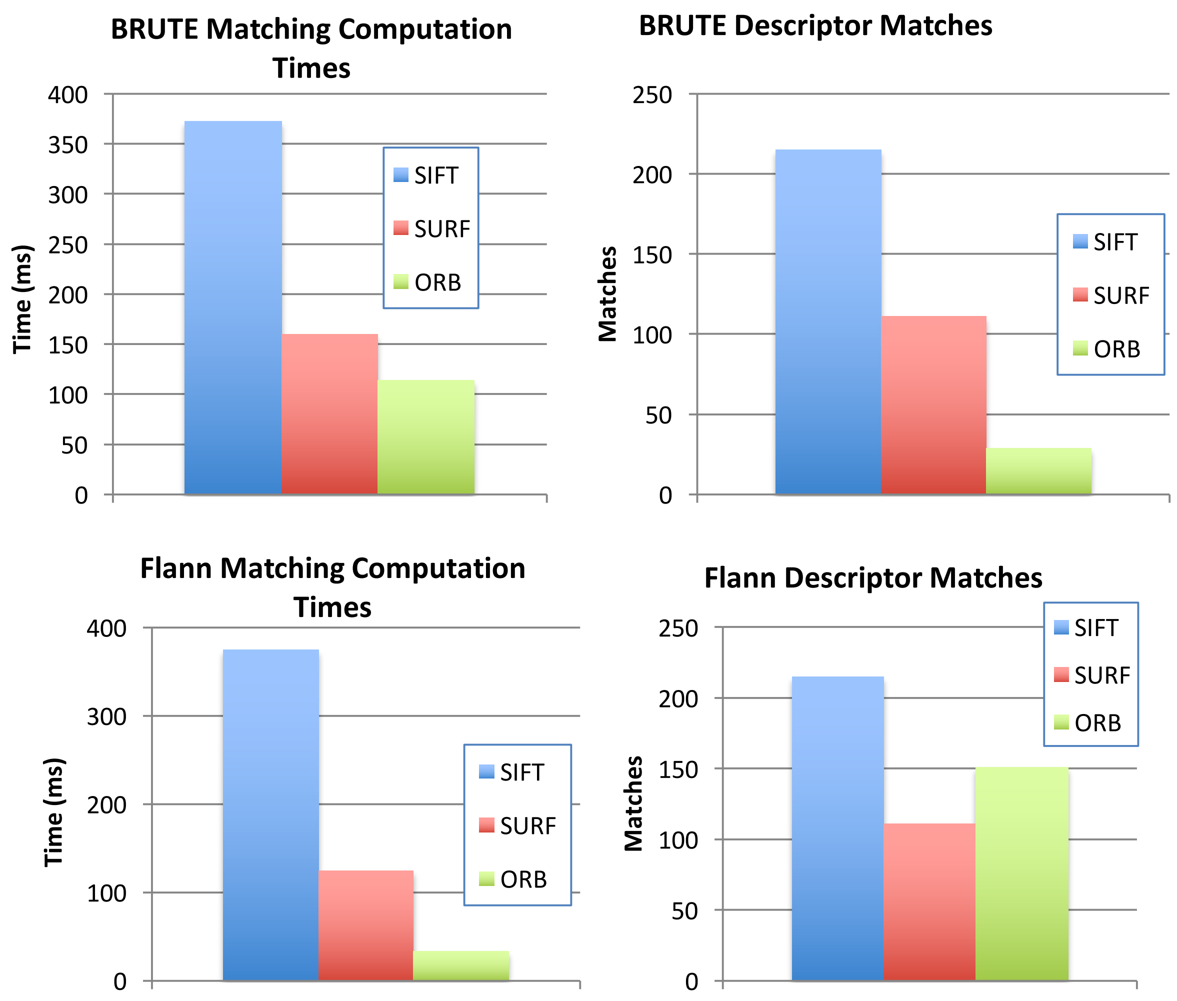


Figure 41. Testing BRUTE and Flann feature matching algorithms with database of truss images.

The resulting data reveals that the ORB feature detector combined with the FLANN Matching algorithm was able to match descriptors between the query and the train images the fastest—with an average computation time of 32.2 milliseconds. ORB-Brute was second-fastest, averaging 111.1 milliseconds and SURF-FLANN was third fastest at 121 milliseconds.

However, despite being among the slowest combinations, the SIFT algorithm with either the FLANN-based or Brute-Force matcher consistently found the most descriptor matches between images--making those combinations the most accurate.

For our application, the ORB-FLANN algorithms were chosen to perform all our descriptor due to the fast computation times and impressive accuracy.

Note that unlike the line detection-based method tested previously, most of the calculations and image processing steps required to perform can potentially be done entirely by the feature detection algorithm—which would greatly speed up calculation time. Unfortunately, feature detection alone is not always reliable given the truss structure. Thus, our final truss detection implementation employs both methods. The line detection method is generally used to find the entire truss within an environment, and feature descriptor matching is performed to detect finer details of the truss such as truss end points and robot gripper grasping points.

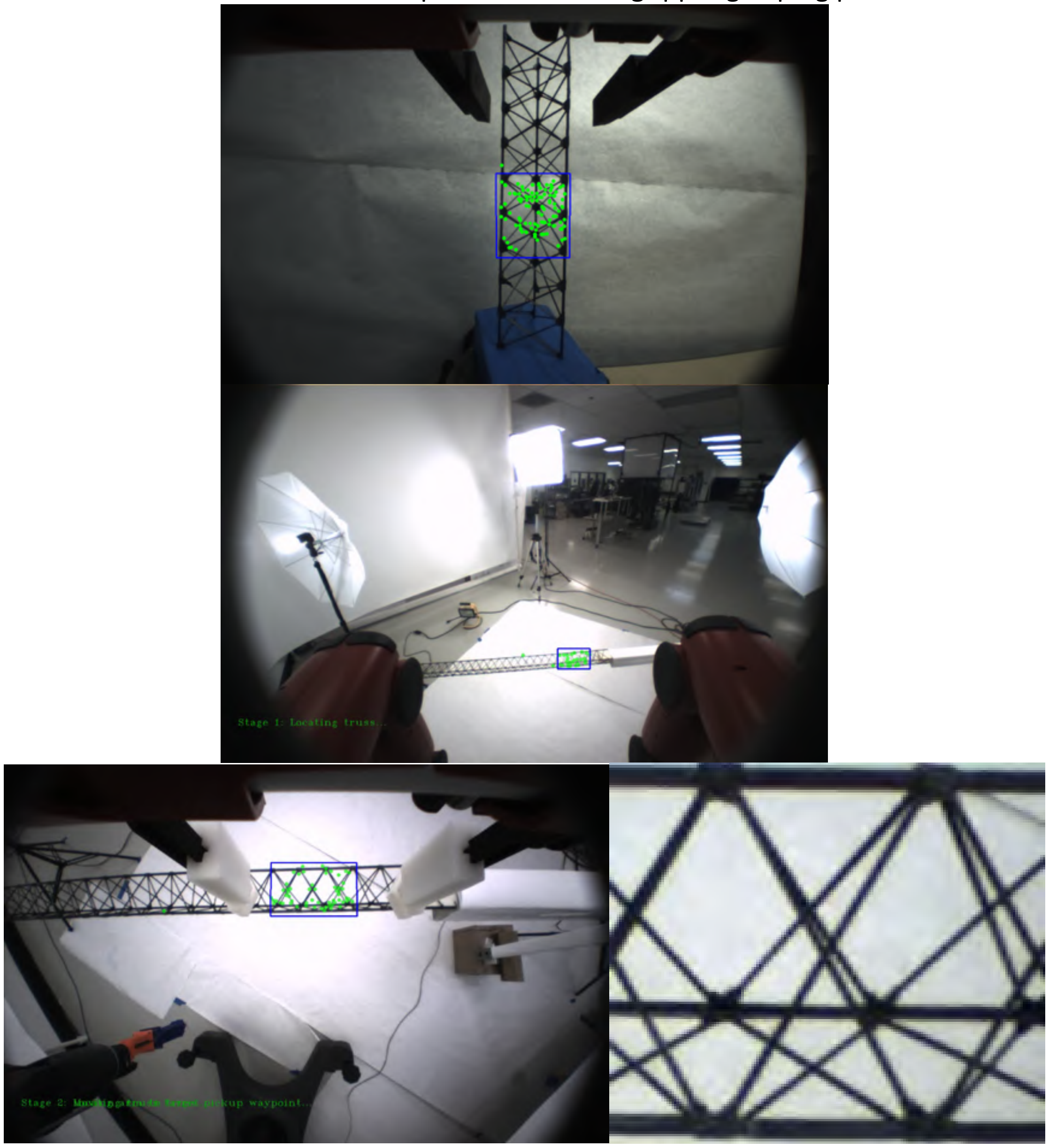


Figure 42. Truss Object Detection using ORB-FLANN Feature Descriptor Matching with a common query image (bottom right).

During our testing, ORB-FLANN feature descriptor matching worked well and generally provided consistent results in various lighting conditions and scales. It did however, struggle when the same query image was used to detect truss at significantly different angles. We chose to compensate for this by using multiple query images of trusses at different angles. If a truss could not be found, we cycled to the next query image until the truss was either identified or presumed absent from the train image.

2.2.4 SpiderFab Robotics Testing Software Platform

To facilitate development and testing of the control algorithms, robotic tools, and assembly CONOPS, we developed a software platform combining GUI, controls, and visualization. Since we expect our robotic software applications will continually evolve as the SpiderFab architecture matures from prototyping stages to final implementation, we decided to structure our software infrastructure to achieve our long-term goals of creating flexible, robust and hardware-agnostic software applications for SpiderFab. This new infrastructure consists of a re-tooled software version control system, improved coding standards, and a new software architecture that will serve as the foundation upon which all future robotics applications will be constructed.

2.2.4.1 Architecture

The software architecture for our robotics applications serves as both a shared blueprint and accelerated starting point for future applications. Because of this commonality amongst applications, a significant amount of the developed code can be reused, thereby speeding up the development process as well as subjecting the code to rigorous functional testing, making it more robust and reliable in the long term.

Care was taken to establish clear design goals for our new software framework to ensure our resulting implementation remained flexible enough to accommodate a wide range of applications and hardware platforms. After researching preexisting software architectures with similar aims, we ultimately decided that it must have the following characteristics:

1. <u>Hardware Agnosticism</u> – All software applications should be portable across various hardware platforms with minimal effort. An example of such an application would be the control software used for commanding robot movements. This application would ideally be able to run on a PC during early testing phases and then later ported to the final embedded hardware.
2. <u>Reusability/Modularity</u> – The architecture should allow for a significant amount of developed code to be easily reusable across multiple applications. Additionally, software code bases and drivers should be structured in a self-contained manner to allow for applications to be flexible to changes in hardware and software requirements.
3. <u>Scalability</u> – Applications should be allowed to start small and then incrementally scale up as the project matures while minimizing unnecessary complexity.
4. <u>Code Quality/Clarity</u> – The code should be well structured, easily understood by developers and reliable. The minimization of code complexity should always be emphasized during code development.

With our design goals now clearly established, our resulting software architecture was designed using a hierarchical approach commonly used by many middleware web services and cross-platform software frameworks (such as the Android mobile operating system).

The idea is to create a collection of software components organized into “layers” of an overall software stack that provide common or platform-specific services that the application requires but does not necessarily want to handle directly. This adds several layers of abstraction between the high-level application software, the mid-level software drivers and low-level hardware drivers—making applications flexible to changes in software requirements and reusable across hardware platforms simply by replacing certain layers of the software stack. This manner of structuring code should allow for all of our target goals to be reached. Below is the generic software stack used for SpiderFab software applications.

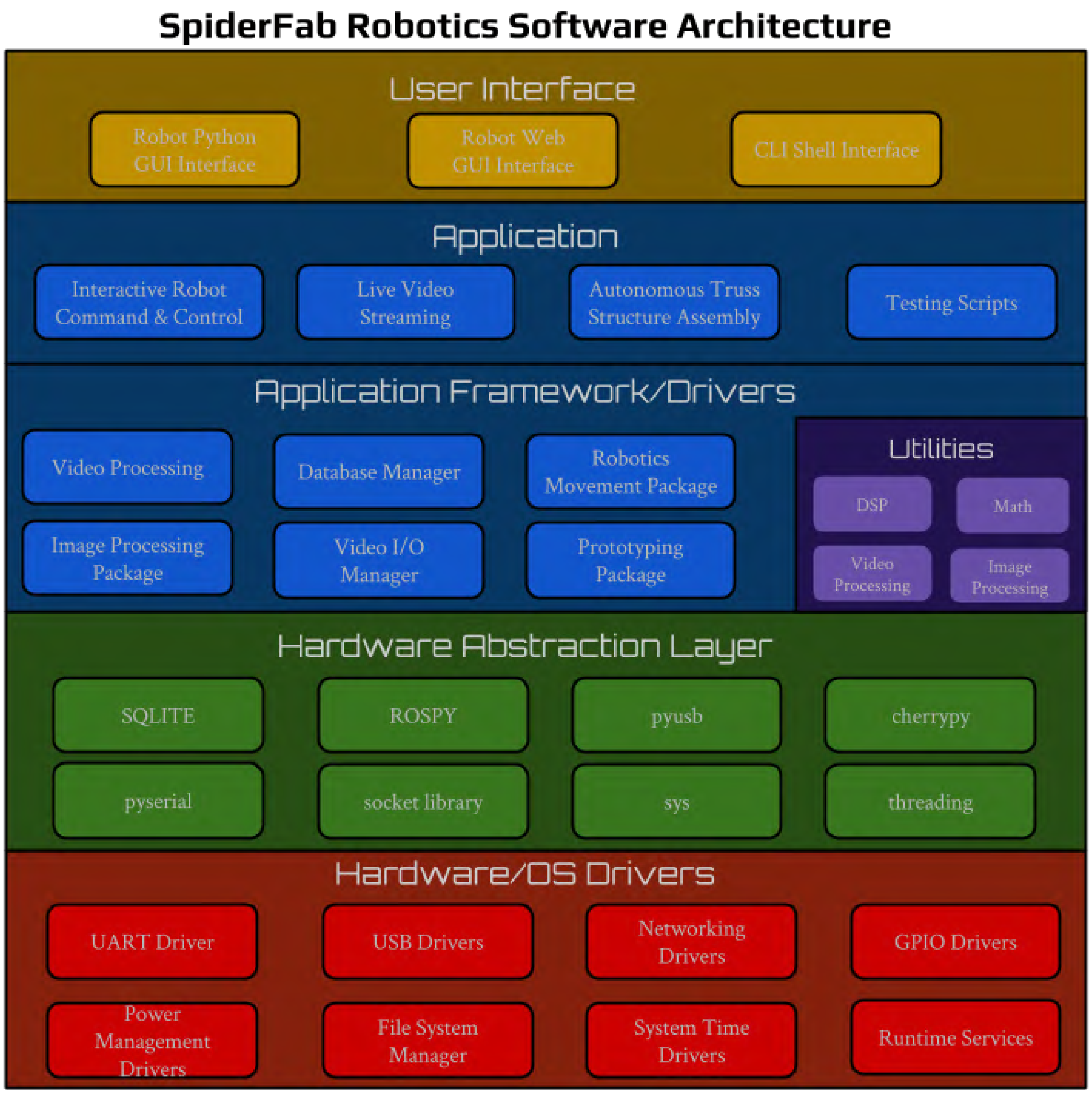
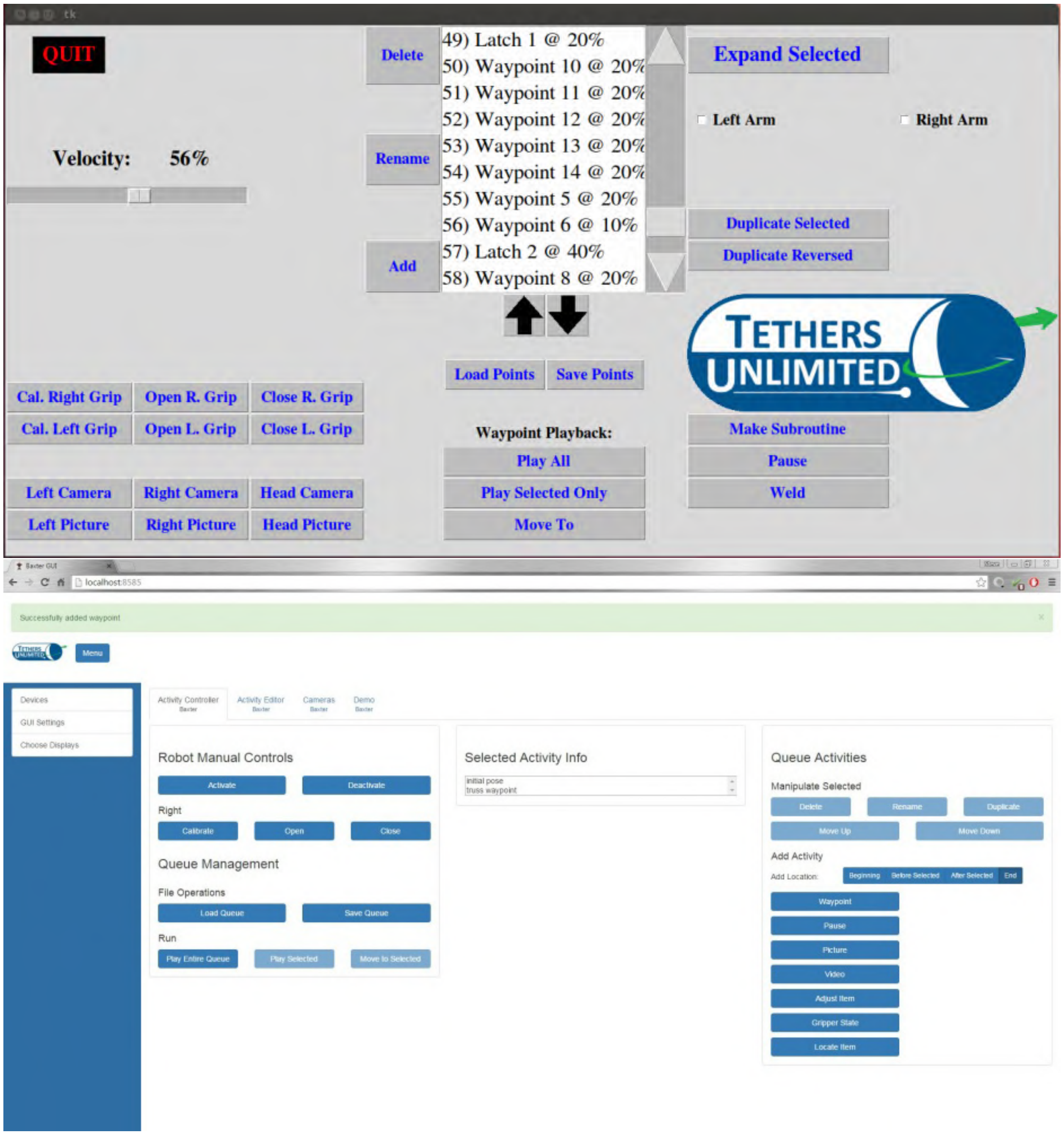


Figure 43. SpiderFab Software Architecture.

1. **User Interface** – Software specifically built to handle the input/output information of the application. A GUI application running on a PC or a command line shell for an embedded device running Linux are common examples. Proper partitioning of user interface code from the application layer should allow applications to run off multiple user interfaces with minimal effort. We developed several user interfaces for our SpiderFab applica-

tions, including Python and web-based GUIs for controlling our robotic test unit Baxter, as shown in Figure 44.



49) Latch 1 @ 20%
50) Waypoint 10 @ 20%
51) Waypoint 11 @ 20%
52) Waypoint 12 @ 20%
53) Waypoint 13 @ 20%
54) Waypoint 14 @ 20%
55) Waypoint 5 @ 20%
56) Waypoint 6 @ 10%
57) Latch 2 @ 40%
58) Waypoint 8 @ 20%

Successfully added waypoint

initial pose
truss waypoint

Figure 44. Example of running an application from two different GUI interfaces. The first GUI (top) was quickly built for initial testing using the Python TK library. We later upgraded to a custom HTML-based GUI (bottom) for a cleaner cross-platform interface with more sophisticated controls.

2. **Application** – This layer contains the main application software. Testing scripts, robot control software and video streaming applications are a few applications created for SpiderFab.

3. **Application Framework & Drivers** – Contains generic driver code that applications can access directly to perform a common task. An example is the image processing libraries that both our video streaming and truss assembly applications use for processing photos taken by our robotic cameras.
4. **Hardware Abstraction Layer** – Wraps low level hardware drivers into a common interface that the framework drivers can use. This allows all the upper layers of the software stack to be reused across hardware platforms simply by modifying code at this level.
5. **Hardware/OS Drivers** – Low level software drivers that directly interact with system hardware. These generally come prepackaged with the hardware and do not have to be developed internally.

Figure 45 below illustrates the software stack of our Baxter robot command and control software running on our custom HTML GUI implemented using our SpiderFab architecture.

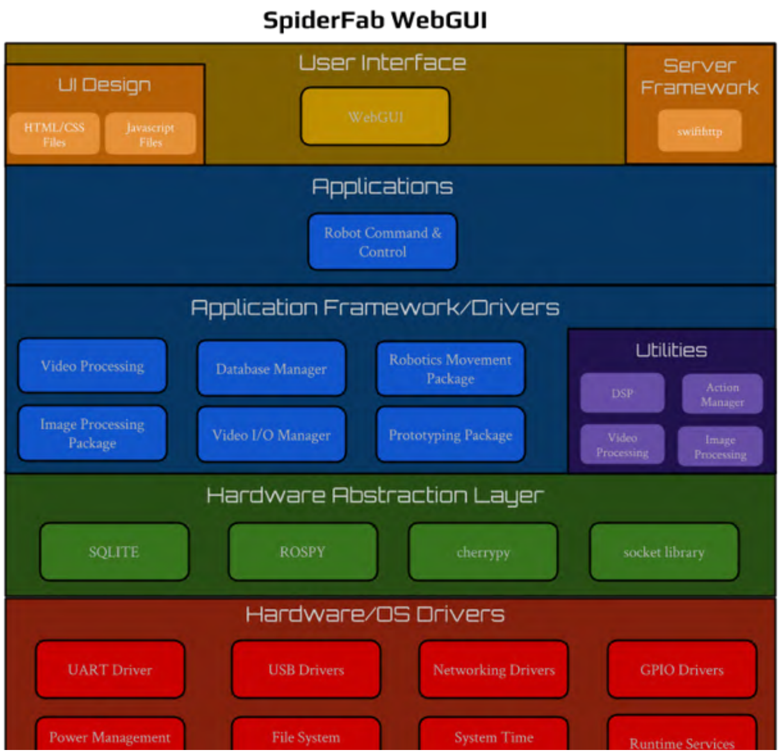


Figure 45. Baxter Command and Control Software Architecture Using Web GUI.

2.2.4.2 Software Infrastructure Upgrades

Once our software architecture was in place, we updated our version control practices to take advantage of this layering separation. Each layer in the SpiderFab Architecture software stack became a separate sub-repository. New applications are built in separate project workspaces, with directories structured in manner similar to our software stack layout. Our coding standards were also optimized to ensure our software applications are of the highest quality. This included the introduction of new tools to facilitate improved code transparency and auto-documentation into our code bases.

2.2.5 Custom End Effector

To enable robotic systems such as the Baxter to grasp and manipulate trusses, we prototyped and evaluated several different gripper designs. The design space looked at trading off compliance, locating features, and size to account for a robotic arm with larger end pose tolerances. It was desired to have a gripper that would locate the truss when the end effector grabbed it and have enough compliance to allow the end effector to be offset from the optimal position.

The base end effector used on the Baxter robot was the parallel gripper, shown in Figure 46. This gripper had 2 fingers that were moved to an open or closed position. There were screw mounts as well as ridges on the fingers to allow for customizing the gripping surface.

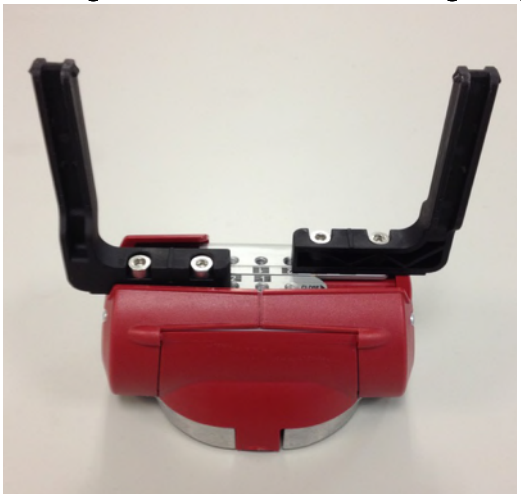


Figure 46. An unmodified Rethink Robotics parallel gripper designed for use with the Baxter robot.

Because the Baxter robot's motions had a measured tolerance of approximately ½ centimeter the gripper designs had to be compliant. It was also desired that when the grippers closed, the truss was put into a known and desired orientation. Figure 47 displays some of the gripper fingers that we tested.

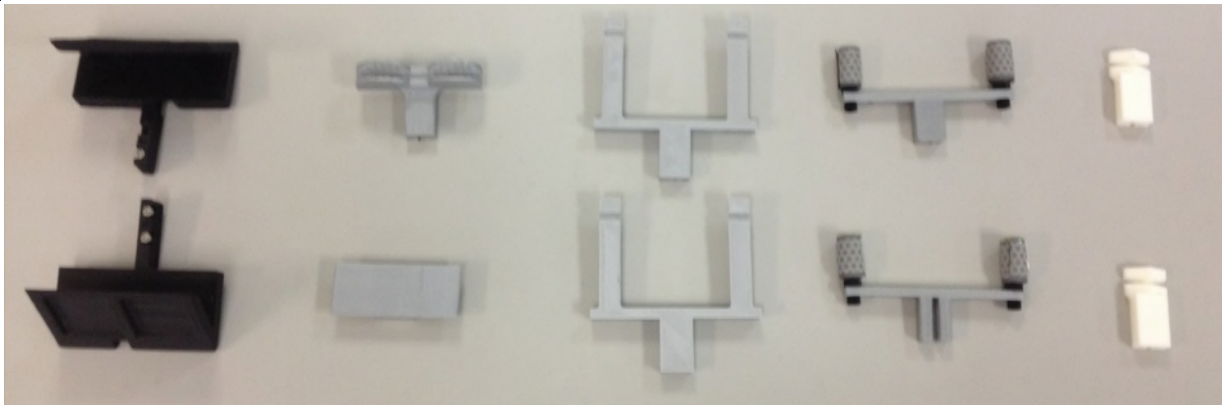


Figure 47. Various designs of gripper fingers.

Initial testing was conducted using an asymmetrical truss and no vision system. To align the truss axially, a set of triangular fingers were tested first. These fingers were bulky and had trouble grabbing the truss off of a flat surface. Another finger design gripped the truss's longeron. This was considerably less bulky than the triangular fingers, but lacked the desired 3 points of contact needed for 6DOF control of the truss. As such, these fingers would often twist the truss into an orientation that was unworkable.

The revised Trusselator we have developed in our Phase II SBIR effort produces trusses that are symmetrical. This enable us to design a finger that gripped the truss on the diagonal members. This provided stability and a compact design. However, these grippers required a much more accurate placement than the previous grippers. To accomplish this precision, we developed the vision system discussed in Section SpiderFab Robotic Vision System. This vision system enabled us to reduce the final position error of the end effector to approximately 3 millimeters. Figure 48 shows one image from the correction algorithm for aligning the fingers to fit between the diagonal members.

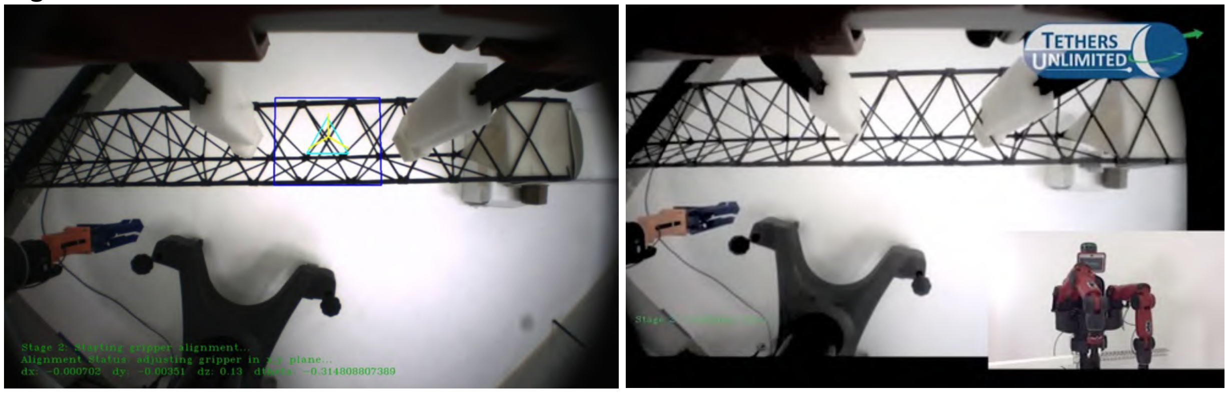

Stage 2: Starting gripper alignment...
Alignment Status: adjusting gripper in x,y plane...
dx: -0.000702 dy: -0.00351 dz: 0.13 dtheta: -0.314808807389
TETHERS UNLIMITED

Figure 48. Robot identifying truss grasping point and then capturing it with the final gripper design.

2.2.6 Truss Assembly Demonstrations

To test our improved software, gripper design, and new vision system, we developed a proof-of-concept demonstration that required our robot test unit (the Baxter Research Robot), to perform the following actions listed below completely autonomously.

1. Identify and locate the truss in our test environment.
2. Estimate the truss pose in 3D space.
3. Pick up the truss using feature detection algorithms to locate a grasp point on the truss, a closed-loop gripper alignment algorithm and our new gripper design.
4. Manipulate the truss and place it in a custom truss joint attached to the other robot arm.

We broke up the demo into three stages. Action items one and two from the list above were stage one of the demo, and stage two and three were responsible for items three and four respectively.

2.2.6.1 Stage 1 – Truss Object Detection and Location Approximation

For the first stage, a live video stream of 1280x800 frames from Baxter's head camera was activated. Once the stream settled to our desired frame rate, we used our new line detection image processing algorithms detailed in section 2.2.1 on each frame received from the video stream. Although this slowed the framerate from 30 frames per second (fps) to roughly 7 to 10 fps, this speed is still more than adequate for performing "real-time" operations. For visual effect and debugging purposes, text and image processing overlays were also added to each frame in real-time, as shown in Figure 49.

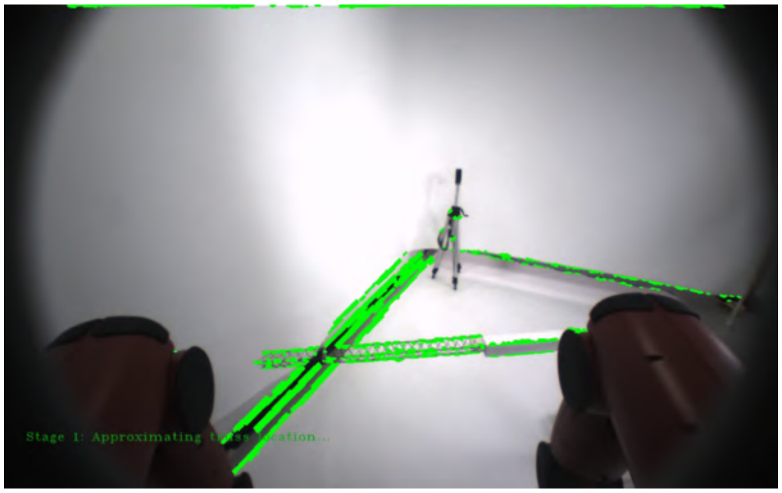


Figure 49. Truss identification using edge and feature detection algorithms.

Once the line detection algorithms detected our truss, we used the ORB-FLANN feature descriptor matching algorithms (now optimized from previous tests) to locate the truss segment that the robot gripper will attempt to use to pick up and manipulate the truss, as shown in Figure 50.

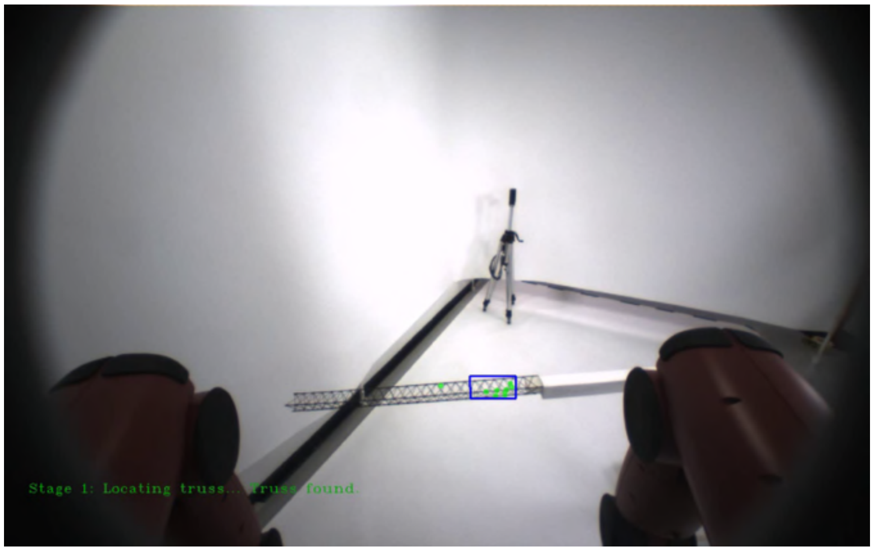


Figure 50. Truss location approximation using homography algorithms.

2.2.6.2 Stage 2 –Truss Alignment and Grasping

Stage two required Baxter to grasp and pick up the truss. This was the most challenging portion on the demo due to the high degree of precision required to align the grippers to pick up the truss and also the lack of precise movement capabilities by Baxter. Given our truss pickup point, and using standard x, y, z coordinates with respect to our gripper cameras (the z coordinates), our grippers has to be aligned with a tolerance of approximately +/- 2 millimeters in the x-plane, +/- 3 millimeters in the y-plane and +/- 3 millimeters in the z-plane. However, before we could begin implementing gripper alignment algorithms, we faced a common, but non-trivial, robotics problem of attempting to specify movements in different frames of reference.

All movement commands sent to Baxter are specified as arrays of joint angles which are generated from an inverse kinematics algorithm using 3-dimensional quaternion coordinates in a "base" frame of reference (where the origin is located at the physical base of the robot). However, when performing the gripper alignment, it is highly desirable to move the grippers using coordinates in the camera's frame of reference.

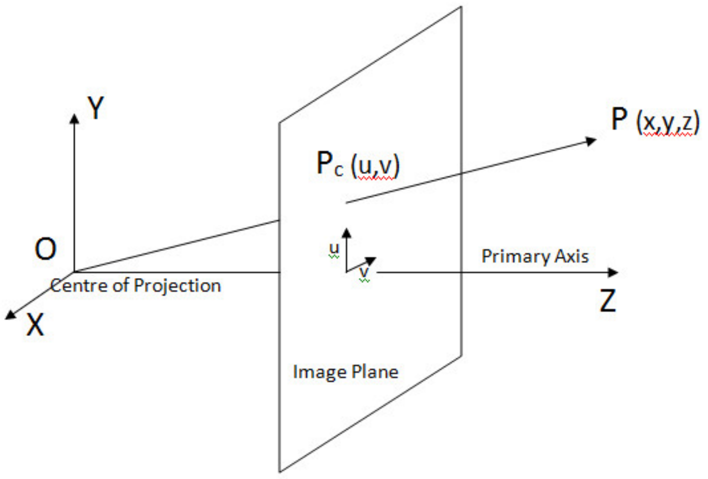
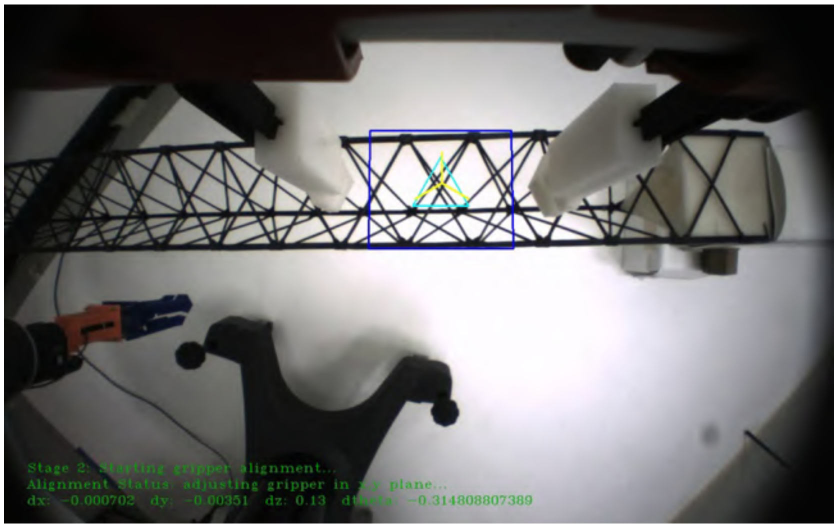


Figure 51. Camera calibration parameters.

Once we could specify coordinates in the camera's reference frame, and the camera calibration factor was known, aligning the gripper could be solved by

1. Identifying a grasping point on the truss.
2. Determining the location on the truss the gripper is currently aligned to.
3. Creating a feedback loop to correct the gripper angle and x, y, z coordinates until it becomes aligned with our target grasping point.

Feature descriptor detection was used to find our grasping point—which for this test we arbitrarily chose to be the spacing adjacent to the fifth longeron crossing from the end of the truss.



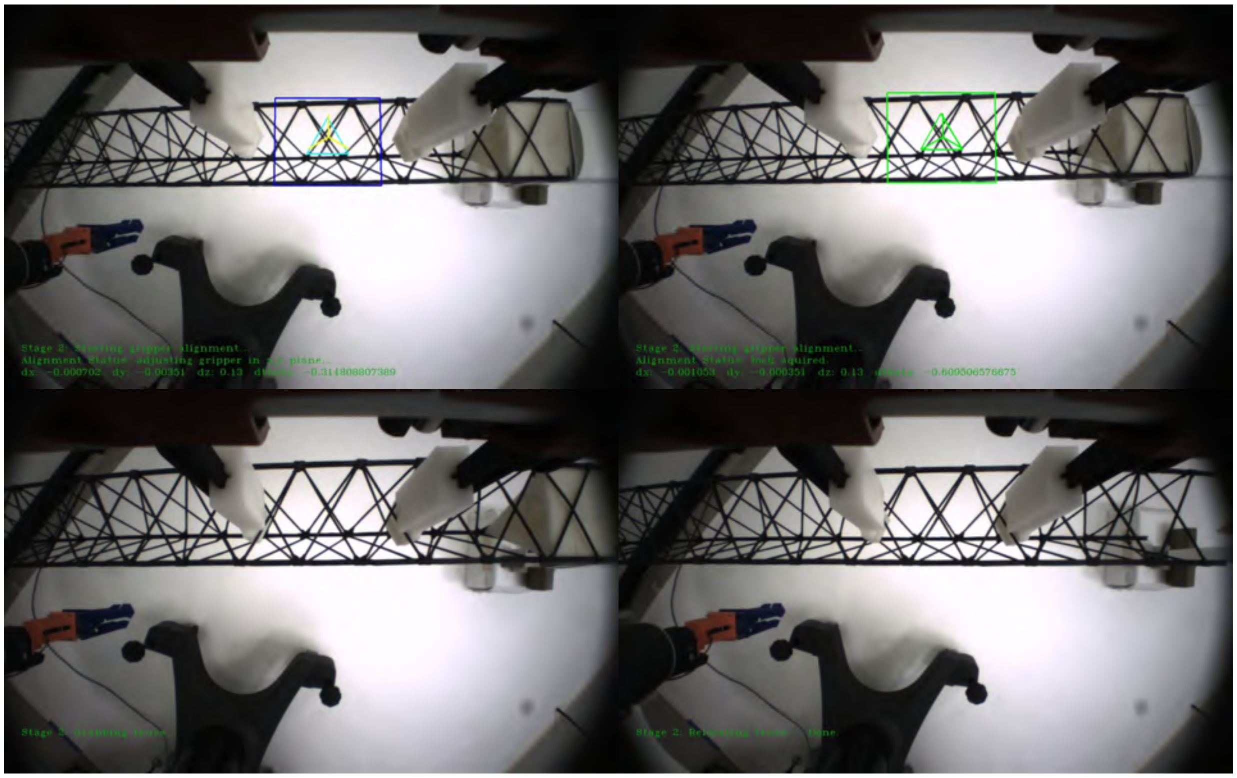
```
Stage 2: Starting gripper alignment...
Alignment Status: adjusting gripper in x,y plane...
dx: -0.000702  dy: -0.00351  dz: 0.13  dtheta: -0.314808807389
```

Figure 52. Robot identifying truss grasping point.

The descriptors generated by the ORB feature detection algorithm contains a variety of information—including orientation information that can be used to discern the orientation of the truss in the x, y plane. It should be noted that because the cameras used by Baxter are not stereovision, it was not possible to determine where the truss was located in the +/- Z direction. That is, we could not determine how far the truss was from the arm camera using only the 2 dimensional images it was generating. This distance information is necessary in order to transform distance measurements from pixels into metric units (i.e. if the gripper is off alignment 20 pixels in the +X direction, we should know how many millimeters in the –X direction to move the robot arm to compensate).

This was a difficult issue to overcome and given our short time frame to complete this demo, we initially decided to use the ultrasonic range sensor located on the robot arm to judge distance. Unfortunately, the data measured by the sensor was often inconsistent due to the lack of solid surfaces on the truss. After many unsuccessful stage 2 attempts, we eventually conceded that we would need to hardcode an expected range in the interest of time--making this one of the few pieces of information that were not generated autonomously by the robot in the demo.

After the distance value was hardcoded to 10-13 centimeters, our algorithms began generating correct gripper adjustment values. Unfortunately, we next ran into an issue with our robot test unit. During our demo, Baxter initially had considerable difficulty executing fine adjustments to the gripper position. The robot would often not respond to commands to move the gripper short distances (which was often less than 3 millimeters) or would overshoot the desired location. Given the small margin for error, this led to very inconsistent grasping attempts. After tweaking the test setup numerous times, we eventually found that gripper movements were much more precise when the robot arm was placed in certain poses during the alignment phase. Once this was taken into account, Baxter was able to complete Stage 1 and 2 with much better consistency.



```
Stage 2: Starting gripper alignment...
Alignment Status: adjusting gripper in x-y plane...
dx: -0.000702  dy: -0.00351  dz: 0.13  dtheta: -0.314808807389
```

```
Stage 2: Starting gripper alignment...
Alignment Status: lock aquired.
dx: -0.001053  dy: -0.000351  dz: 0.13  dtheta: -0.609506576675
```

```
Stage 2: Grabbing truss...
```

```
Stage 2: Relocating truss... Done.
```

Figure 53. Real-time closed-loop grasping of truss.

2.2.6.3 *Stage 3 – Truss Joint Assembly*

The final stage required Baxter to maneuver the truss and place it into our 3D printed truss joint, which was designed to be attached to the end of the truss by a custom end-effector.

Since both the end effector truss joint and the truss (assuming it was grabbed in the correct location) are at known locations and dimensions, placing attaching the truss to the joint was simply a manner of aligning both arms such that the two objects could joined. In the future, this alignment process will be aided by the image processing algorithms using the robot head camera.

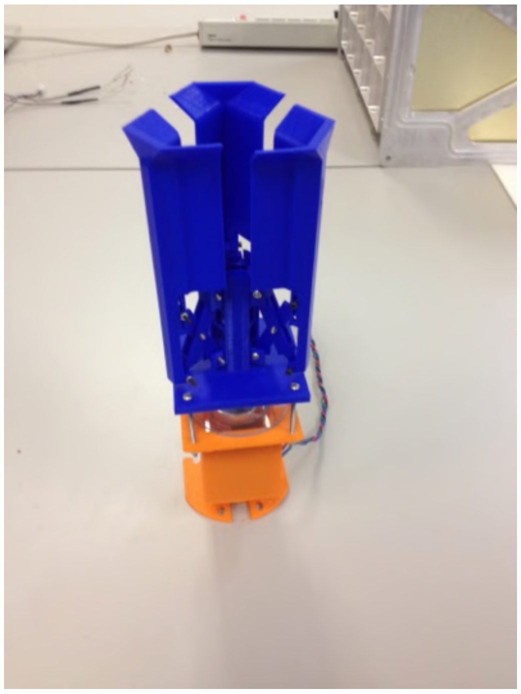


Figure 54. Truss-joint and end effector.

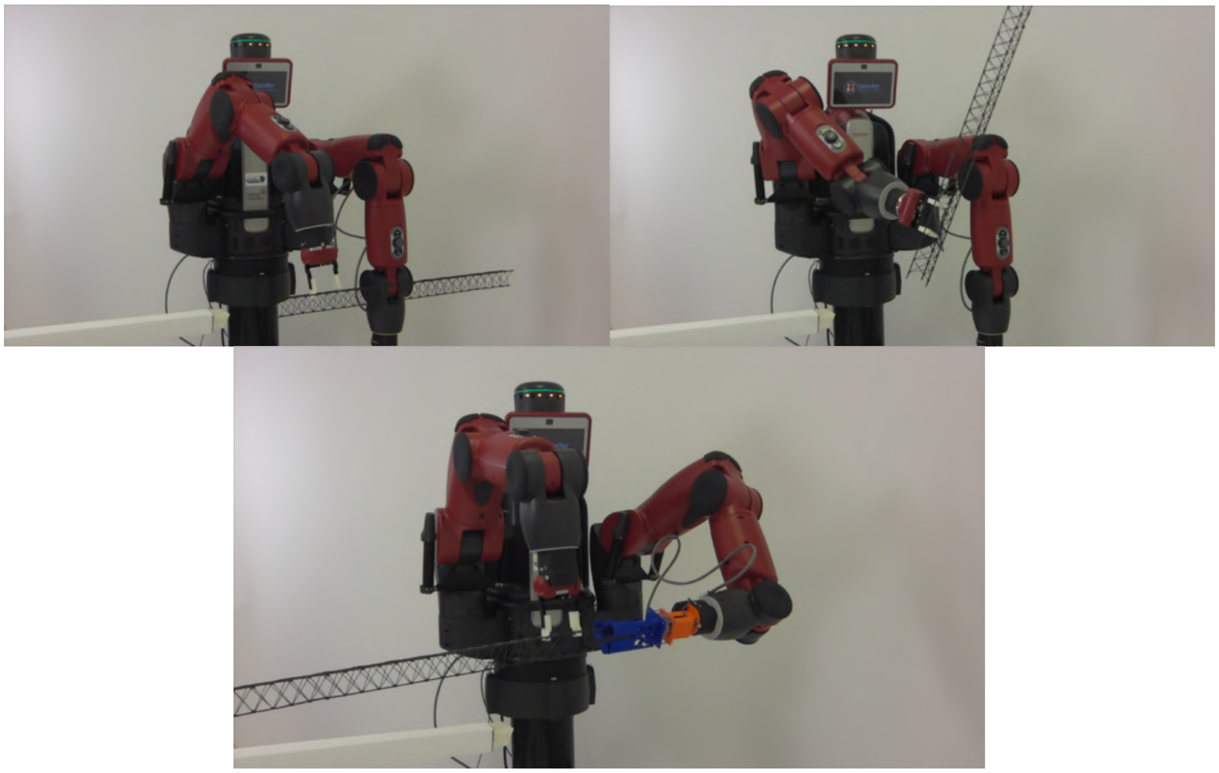


Figure 55. Truss to custom joint end effector assembly.

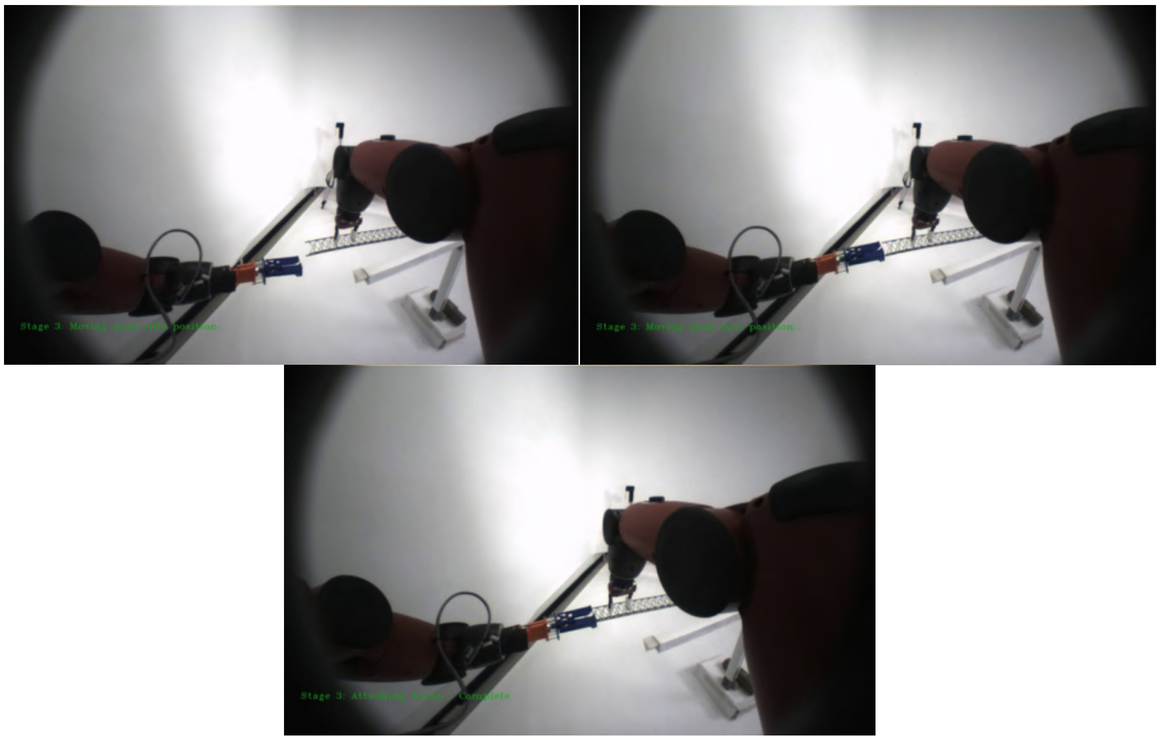

Stage 3: Moving joint into position...
Stage 3: Moving joint into position...
Stage 3: Attaching truss... Complete.

Figure 56. Truss to custom joint end effector assembly (head camera).

2.2.6.4 Demonstration of Robotic Assembly of Spanner Joints

Although the joint assembly demonstration described in Section 2.2.6.3 is relatively straight-forward and would enable large structures to be assembled using a truss-and-connector meth-od, similar to a 'tinkertoys' assembly approach, it has the disadvantage that the mass of the joint connectors will likely dominate the structure mass. A far more efficient structure can be constructed using spanner joints to connect between nodes in the truss, as was discussed in 2.1.1.3. Accordingly, we performed demonstrations using the Baxter robot wielding the Joiner Spinneret described in Section 2.1.4 to validate the feasibility of robotic systems assembling trusses using the highly efficient spanner joint method, as shown in Figure 57.

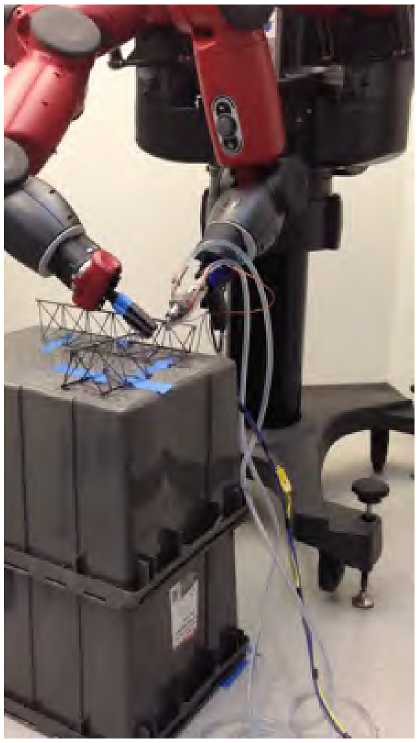


Figure 57. Demonstration of assembly of a spanner joint between perpendicular truss segments using the Joiner Spinneret end-effector on the Baxter robot.

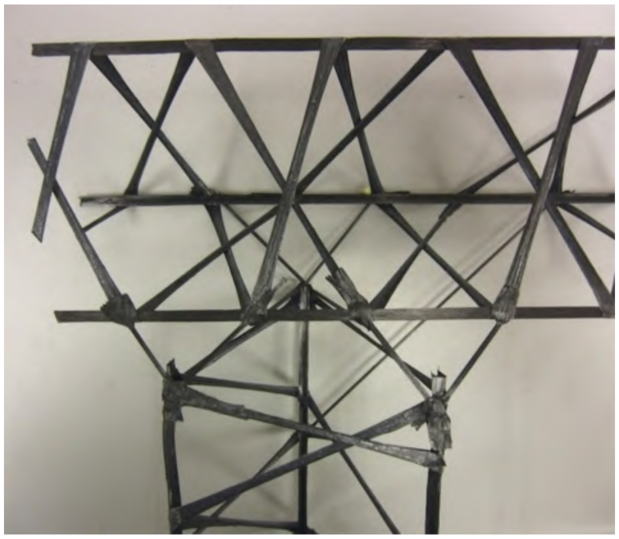


Figure 58. Spanner Joint Geometry. *Highly efficient joints can be formed between truss elements using the Joiner Spinneret to add additional CF/PEEK ligaments between nodes on the truss..*

2.3 Demonstration Mission Concept Designs

2.3.1 MakerSat-I: Long-Baseline Sensor Mission Concept

In order to perform a low-cost initial demonstration of the value proposition of in-space manufacture of space structures, TUI is currently designing a CubeSat-based flight demonstration, called MakerSat-I. The primary objectives of MakerSat-I are to to demonstrate ISM of a large space structure, characterize its structural performance, and demonstrate the utility of a Constructable structure as part of a long-baseline sensor system.

The preliminary configuration concept for MakerSat-I, shown in Figure 59, is configured as a 6Ux1U CubeSat for compatibility with the NanoRacks deployer aboard the ISS; a 2Ux3U configuration compatible with the CSD and other 6U deployers is also feasible. The MakerSat-I system will integrate our 3U Trusselator system with COTS CubeSat command and data handling (C&DH) and electrical power system (EPS) as well as a HYDROS water-electrolysis thruster and two SWIFT-XTS X-band software defined radios (SDRs), one positioned at and end of the truss system. An optical fiber dispenser derived from TUI's Underwater Optical Fiber Dispenser will also be integrated into the system, and highly sensitive accelerometers will be integrated at both ends of the system.

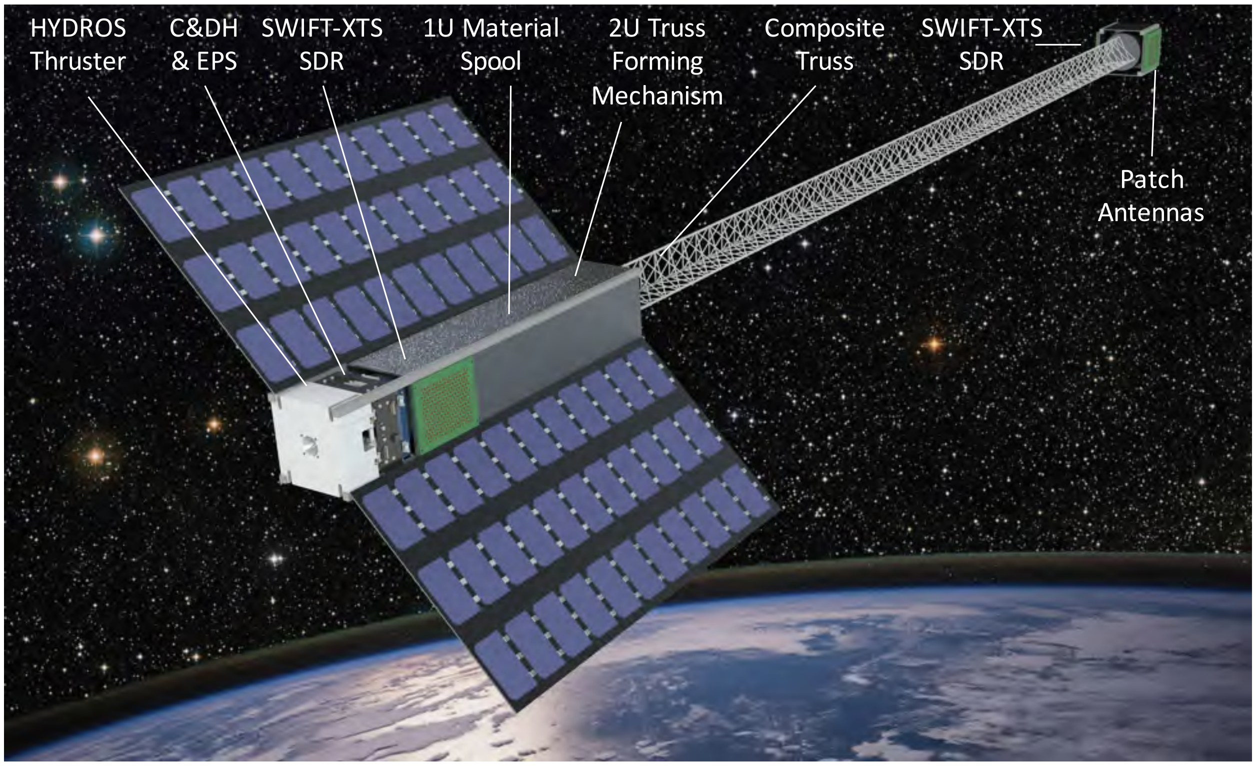


Figure 59. MakerSat-I concept 6U configuration. *The 6U MakerSat-I CubeSat will use a 3U Trusselator system to fabricate a 50m truss in between two X-band software defined radios to demonstrate long-baseline one-pass Interferometric SAR capabilities.*

In the baseline MakerSat-I mission concept, after deployment from the ISS, the system will first deploy its solar panels and antennas. After a delay sufficient to ensure safe separation from the ISS, the satellite will use its HYDROS thruster to move to its desired operational orbit. The system will then activate the Trusselator system, which will additively manufacture a 50m truss. To provide a sense of the scale involved, Figure 60 shows the 50m truss juxtaposed with the

SpaceX Dragon capsule. As the system fabricates the truss, the fiber dispenser will deploy the fiber inside the truss, running from one end of the truss to the other in order to provide high-bandwidth data transfer and sub-ns synchronization between the two SDRs. This timing synchronization will enable the SWIFT-SDRs to operate coherently and provide for phase alignment between the radios. As the system manufactures truss, the truss integrity will be diagnosed by forcing vibrations into the system using thrust impulses provided by the HYDROS thruster and/or small vibratory motors, such as those used in cell phones, and recording the response of the truss structure using the accelerometers positioned at both ends of the system.

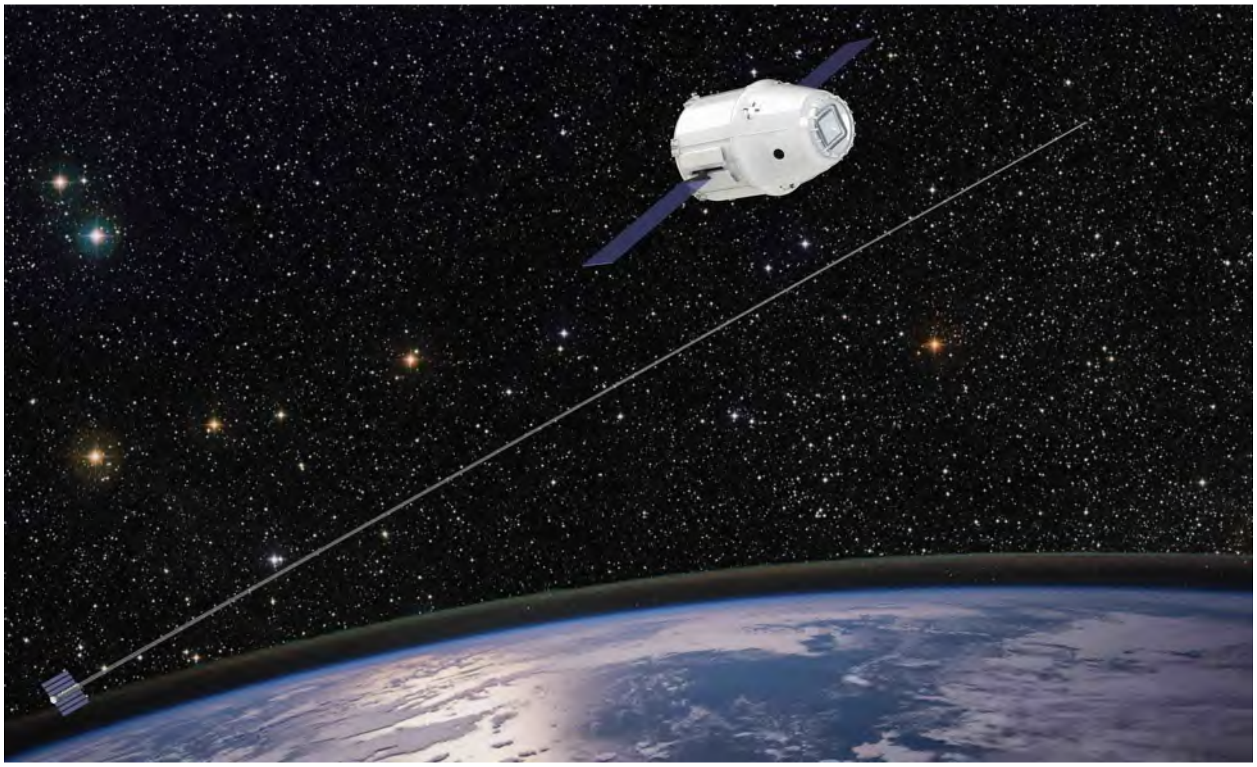


Figure 60. MakerSat-1 "Constructable™" Long-Baseline Sensor. *Dragon capsule shown to provide a sense of scale.*

To demonstrate the utility and value proposition of this Constructable structure, the MakerSat-I mission will then operate the two X-band SDRs as an interferometric synthetic aperture radar (InSAR) or as a long-baseline astronomical interferometer. For InSAR demonstrations, the MakerSat-I system will operate parasitically, relying upon radar transmissions by a much larger satellite. As the truss deploys, gravity gradient forces will tend to orient the system along the local vertical, but off-vertical orientations can be achieved to improve InSAR performance by using the HYDROS thruster to set the system into a cross-plane libration.

The potential performance of this small, low-cost InSAR system can be estimated using the expression for the sensitivity of InSAR height measurements to phase errors:[2]

$$|\partial h| \approx \frac{\lambda R \cos\psi}{4\pi \,\mathrm{B}\sin(\psi+\beta)} |\partial\phi|, \tag{1}$$

[2]. Richards, M.A., "A Beginner's Guide to Interferometric SAR Concepts and Signal Processing", IEEE A&E Systems Magazine, Vol. 22, No. 9 September 2007

where λ is the RF wavelength, ψ is the angle between the local horizontal and the target, β is the angle between the baseline and the local horizontal, B is the baseline length between the receivers, and $\partial\phi$ is the phase measurement error between the receivers. The SWIFT-XTS SDRs are designed to enable coherent operation, and testing under prior efforts has indicated that phase coherency within at least 10 degrees, and potentially as low as 1 degree, is possible. With a baseline length of 50 meters, an operating frequency of 9 GHz, an altitude of 300 km, the truss oriented 45 degrees away from vertical in the cross-plane direction, and a look angle of 45 degrees, we expect the MakerSat-I CubeSat could achieve measurement height accuracy comparable to the 6m demonstrated by the ($200M+) SRTM experiment flown on the Shuttle *Endeavor* in 2000, and potentially as low as 1 m in a best-case scenario.

2.3.2 MakerSat-II: Constructable™ Antenna Demonstration Mission

The MakerSat-II mission will demonstrate the technical feasibility of combining additive manufacture of RF reflectors with robotic assembly technologies to enable a compact, lightweight payload to perform in-space manufacture (ISM) of large antennas to enable challenging signals intelligence (SIGINT) and satellite communications (SATCOM) missions.

Motivation: Currently, multiple commercial and government efforts are pursuing development of constellations of small satellites in LEO to provide high-bandwidth, low-latency, resilient data services to ground users. Due to the small sizes of the RF apertures these SmallSat systems can deploy, closing the data link requires that the ground users connect to satellite terminals or 'hotspots' having antennas at least the size of a laptop computer. This requirement for a bulky and expensive antenna limits the potential market for such a system. A Direct-to-Smartphone broadband (DTSB) service would dramatically increase the utility of a SATCOM system for consumers and military operations. However, closing a broadband (≥10 Mbps) data link from LEO to a smartphone carrying an omni antenna and having ~33dBm transmit power will require the satellite use a high gain antenna with diameter on the order of 10-25 meters. Although high-TRL large deployable antennas are available, they have very high recurring costs (~$500K/m^2), and even when stowed require a large volume within a launch shroud. Consequently, fielding a DTSB SATCOM system using current Deployables technology requires high launch and recurring satellite fabrication costs, making DTSB systems financially untenable unless a more affordable large-antenna technology emerges.

Approach: The MakerSat-II effort will design, prototype, and test a compact "Constructable™ Antenna" payload capable of in-space manufacturing of large antenna reflectors. Figure 61 illustrates a palletized work cell that will fabricate a zero-CTE composite truss support structure and concurrently manufacture and attach reflector segments to assemble a large reflector. To construct a large parabolic dish, the system will first assemble the central portion of the dish and then then build up both the support structure and reflector. Fabrication of the support structure will use the Trusselator™ system developed under NASA SBIR contract NNX14CL06C as well as the real-time vision-based robotics control methods and assembly tools discussed in Section 2.2. A metrology system and adjustable mounting features on the truss will enable closed-loop control to achieve the shape accuracy necessary for the antenna reflector.

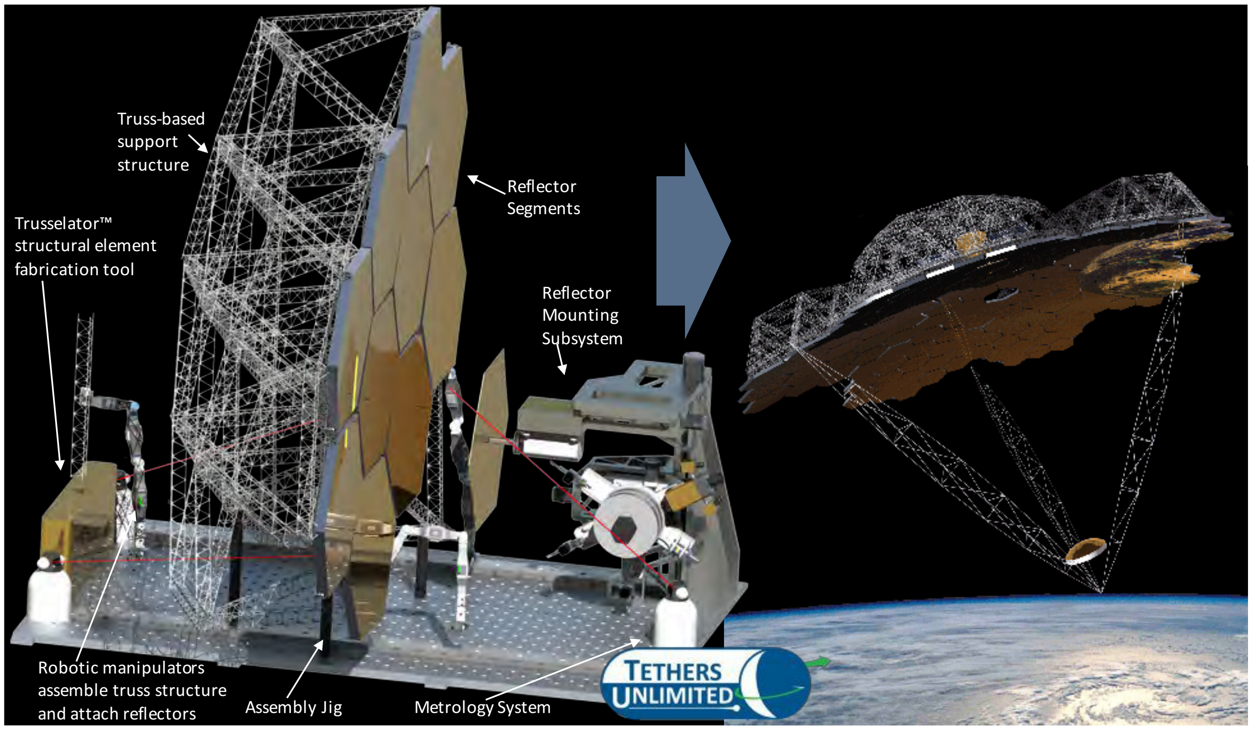

Truss-based support structure
Trusselator™ structural element fabrication tool
Reflector Segments
Reflector Mounting Subsystem
Robotic manipulators assemble truss structure and attach reflectors
Assembly Jig
Metrology System
TETHERS UNLIMITED

Figure 61. Concept for a Constructable™ Antenna payload.

3. OUTREACH AND TRANSITION EFFORTS

3.1 Outreach

In addition to our technical work, we have also performed significant efforts to disseminate the results of our NIAC work as well as to identify potential avenues for post-NIAC transition. Below is a list of major outreach efforts performed during the SpiderFab Phase II effort:

- Presented an invited talk on SpiderFab at the Additive Aerospace Summit in Los Angeles, CA on 17 October 2013.
- Presented SpiderFab to the NRC Committee on Space-Based Additive Manufacturing (COSBAM), Irvine CA 12 November 2013. This briefing resulted in the COSBAM report on 3D Printing in Space (http://www.nap.edu/catalog.php?record_id=18871) highlighting "Creation of Structures Difficult to Produce on or Transport from Earth" as one of the more promising potential applications of additive manufacturing in space.
- Presented an invited talk on SpiderFab at the IdTechEx "3D Printing Live!" conference in Santa Clara, CA on 21 November 2013;
- Dr. Hoyt participated as a subject matter expert in an 'industry ecosystem' workshop on 3D printing at Dupont on 2 December 2013;
- Presented invited talk on SpiderFab at the Additive Disruption Summit in San Francisco, 26 March 2014;
- Presented invited talk on SpiderFab at the WA State Joint Center for Aerospace Technology Innovation (JCATI) Symposium at WSU, 21 April 2014;
- Presented invited talk on SpiderFab application to SBSP at the SolarTech Conference, 23 April 2014, in NYC;
- Presented "SpiderFab: Architecture for On-Orbit Manufacture of Large Aperture Space Sys-

tems" FISO Telecon Colloquium, 4 March 2015;
- Presented SpiderFab at the 2015 NASA Tech Day on the Hill event, 29 April 2015;
- NASA 360 TV story on the NIAC SpiderFab effort: https://goo.gl/996C1F
- Presented a talk titled "It's only Science Fiction... Until You DO it!", featuring SpiderFab, at the "Science Fiction/Science Fact" event at the Museum of Flight, 25 October 2015;
- Presented invited talk on SpiderFab to the Seattle Futurists Society, 12 Dec 2015;
- Presented invited talk on In-Space Additive Manufacturing of Spacecraft Structures at the University of Washington Collaborative Center for Advanced Manufacturing (CCAM), 14 January 2016.

These efforts, along with the (very much appreciated) efforts of Kathy Reilly and others at NIAC to spread the word about our effort have resulted in a large number of positive print and web articles about SpiderFab.

3.2 Transition Successes

As discussed previously, this NIAC Phase II effort very quickly transitioned to post-NIAC efforts in the NASA/LaRC "Trusselator" SBIR contract.

We believe the results of this NIAC effort also contributed to NASA STMD including in-space manufacturing as a technology of interest in the 2015 Tipping Point Technologies program solicitation. That in turn resulted in TUI teaming with a large prime contractor on a successful proposal to perform a demonstration of in-space manufacture of a key GEO satellite structure, and a contract to begin preparing that flight demonstration is pending.

CONCLUSIONS

In-Space Manufacturing (ISM) of key space system components such as antennas, arrays, and optical systems offers the potential to enable NASA, DoD, and commercial space programs to escape the limitations of rocket shroud volumes and create systems with order-of-magnitude improvements in performance-per-cost relative to current state of the art. The SpiderFab Phase II effort made significant progress in validating the technical feasibility of an ISM architecture in which large apertures will be fabricated *in-situ* using techniques adapted from additive manufacturing (3D printing), automated composite layup, and robotic assembly. The effort developed and demonstrated end-effector tools, vision-based control software, and concepts of operation (CONOPS) to enable robotic systems to assemble composite truss elements manufactured in space to construct large, extremely-high-performance support structures for antennas and other apertures. The effort also developed concepts for several affordable technology demonstration missions that will validate the key technologies in a staged, incremental manner. Most significantly, it has resulted in successful transition to post-NIAC activities, including a SBIR contract to develop system for in-space manufacture of truss structures and a NASA Tipping Point Technologies subcontract effort to demonstrate ISM of a key GEO satellite structure.