

**Path Planning for Human Planetary Surface
Exploration in Rough Terrain**

by

Johannes J. Norheim

B.S. Aerospace Engineering, Massachusetts Institute of Technology
(2016)

Submitted to the Department of Aeronautics and Astronautics
in partial fulfillment of the requirements for the degree of

Master of Science in Aeronautics and Astronautics

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2018

© Massachusetts Institute of Technology 2018. All rights reserved.

Signature redacted

Author

Department of Aeronautics and Astronautics

Signature redacted May 21, 2018

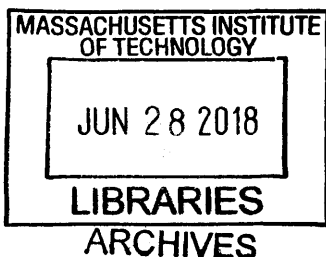
Certified by

Jeffrey A. Hoffman
Professor of the Practice of Aeronautics and Astronautics
Thesis Supervisor

Signature redacted

Accepted by

Hamsa Balakrishnan
Associate Professor of Aeronautics and Astronautics
Chair, Graduate Program Committee





77 Massachusetts Avenue
Cambridge, MA 02139
<http://libraries.mit.edu/ask>

DISCLAIMER NOTICE

Due to the condition of the original material, there are unavoidable flaws in this reproduction. We have made every effort possible to provide you with the best copy available.

Thank you.

The images contained in this document are of the best quality available.

Path Planning for Human Planetary Surface Exploration in Rough Terrain

by

Johannes J. Norheim

Submitted to the Department of Aeronautics and Astronautics
on May 21, 2018, in partial fulfillment of the
requirements for the degree of
Master of Science in Aeronautics and Astronautics

Abstract

This thesis discusses updates to the Surface Exploration Traverse Analysis and Navigation Tool (SEXTANT) that allowed it to be applied in a real EVA path planning and navigation scenario for the first time since pioneering work was done on such a tool back in 2001.

SEXTANT has been ported from MATLAB to Python, which in contrast is an open-source and non-proprietary programming language, an effort which was motivated by NASA's BASALT (Biologic Analog Science Associated with Lava Terrains) campaign. This thesis also discusses its integration with a larger EVA planning, execution, and analysis environment developed at NASA Ames: xGDS (exploration Ground Data Systems) in contrast to the standalone tool that previously existed. It also details the performance of SEXTANT's path planning capability for the first time and introduces new parameters that allow for the trade of time complexity and optimality of the path. The thesis also outlines a method that can overcome problems that were encountered with rough terrain captured in high resolution digital elevation maps (DEM). Next, the thesis discusses results and lessons learned from using SEXTANT during two BASALT deployments in basaltic terrains in Idaho and Hawaii. Finally it goes into details on interfaces that were built to complement SEXTANT: a web-based interface for both planning and navigation purposes, and the Holo-SEXTANT Augmented Reality (AR) navigation solution. The first was used as the primary navigation method for the analog astronaut crew during one of the deployments, while Holo-SEXTANT was used by the analog crew as a technology demonstrator during two simulations, and several times out-of-simulation in the same environment.

Thesis Supervisor: Jeffrey A. Hoffman

Title: Professor of the Practice of Aeronautics and Astronautics

Acknowledgments

I owe my gratitude to many people for the work contained in this thesis. To start, to Jeff Hoffman, who was also my undergraduate advisor, and who was the one that took me on to the BASALT project(that you will hear more about in the thesis) while I was trying to figure out my funding situation. If there is one thing I learned through the past two years is that funding cannot be taken for granted, and Jeff always did his best to make sure I could find a funding source. Thank you for being so supportive, and for not giving me a hard time, when the thesis slipped from “Oh, that can easily be done in a year” to “ok, early spring graduation” to “I better get this written up to graduate on time”. Thank you also for always being supportive of innovative ideas, and for giving the HoloLens a try. It was definitely worth it. While we are in the realm of MIT and people involved in this work let me also reach out a big thank you to Eswar and Jesslyn, my undergraduate researchers for whom I owe a lot of the content in Chapter 5. It was a pleasure to work as a part of a team on something “shiny” like the HoloLens, and going through the adventures we had, both on campus and at Hawaii. I’m so proud on how far you made it, poster presentation, lab presentation, conference paper, all the thing I wish I had been exposed to when I was a UROP myself back in the days. I hope you enjoyed having me as your mentor, for me it was an honor. Finally, thank you Dava for supporting us, getting in touch with the OpsLab people at JPL and making the right connections to move things forward.

Over to the BASALT crew! Wow, never thought I would have met such a diverse team. A huge thank you goes first to Tamar, not only on the professional level, but also on the personal one. For life advice and those things. For the chocolate. And all the other things. Happy things worked out so that we could still interact at regular intervals, even when work didn’t require it. Also a big thanks to Dave, your good mood keeps many of those around you going. Including me during our late night coding at Hawaii. And thanks to Jessica for giving some insights into the past, and the origins of SEXTANT. Who would have thought that after she started working on

this in 2007, destiny would have her working with someone who continued her work? Thank you Darlene, for your warmth and welcoming. For immediately including me as part of the BASALT crew. Also thank you for being so patient, and for setting off time to discuss research ideas. Thank you to the rest of the operations crew, Kara, Steve, Andrew... and to the science team, Shannon, Allyson, Scott, Sam... you taught me to really appreciate the science that gets done in the field from both the geology and the biology end. A big thank you also to Matthew Miller, for taking time to discuss novel ideas and how to shake the new into the old that unfortunately still permeates a lot of NASA.

Back to MIT - I owe a few thanks to the people that make my life exciting and keep me from living in front of the computer and digging myself into a rabbit hole with my research. First to Berk and Elise, the best friends - and flatmates - one could ever wish for. Thank you for keeping me sane through our dinner traditions which would highly motivate me to make it home eventually. For the entertaining discussions, the morning routine and forgetting to plug in the toaster. All these small moments made the stress of MIT so much more bearable. Thank you to Jerome, Elena, June, Sylvain, for inviting me out on hikes, organizing fancy dinner nights, and for the many BSO concert nights. Thank you Sylvain for convincing me to run a 24 hour relay race the weekend before my thesis is due. My ankle will certainly not regret it (sarcastic voice).

Big thank you to the 409 crew - I should technically had an office in the MVL, but managed to get a window view of Mass Ave that I wouldn't exchange for any other office space. You kept my motivation for working on space systems while I would still work on doing the work I was actually responsible for, i.e. writing up this thesis. Thank you also Scott Nill, for providing me with funding for my last semester - even though the topic was completely outside my area of expertise.

Skiing has become a really important part of my life, so thank you to the ski team - for letting me coach you, and giving me some moral support during quals studying times.

Of course a big thank you to my family, for always supporting me, especially my

brother for coming up with amazing trips and coming over to coach the ski team with me and my sister for always providing me good life advice.

Finally, a special thank you to someone that has started taking up a very important part in my life lately. Who would have thought that a trip to Russia through a program few have heard of, would eventually lead to adventures across the US and beyond? Thank you for being patient with me, but also for pushing me to get moving when my internal stress kept telling me to stay home and look at my screen to feel productive. Thank you for being such an important part of my life, and supporting me all the way!

Contents

1	Introduction	17
1.1	Motivation	17
1.2	Thesis Contribution	21
1.2.1	A Formalization of Automated EVA Traverse Planning	21
1.2.2	Performance Evaluation of an EVA Planning Algorithm	21
1.2.3	Automated Traverse Planning Limitations	22
1.2.4	Experience from an Analog Use Case Environment	22
1.2.5	An Open Source, Free, User Friendly Code Base	22
1.2.6	A Flexible Interface and Integration with Several Tools	23
1.3	Thesis Outline	23
2	Background	27
2.1	Context of EVA Planning	27
2.2	The Path Planning Problem	29
2.2.1	Areas of application	29
2.2.2	Problem definition	30
2.2.3	Problem Representations	31
2.2.4	Path Planning Algorithms	33
2.3	SEXTANT background	34
3	Path Planning in Rough Terrain	37
3.1	Path Planning Problem Formulation	37
3.1.1	Waypoints	38

3.1.2	Environmental Description of the Terrain	39
3.1.3	Obstacles	40
3.1.4	Path Planning Metrics	42
3.2	Path Planning Algorithms	45
3.2.1	Original A*	45
3.2.2	Heuristic Function	47
3.2.3	Shortcomings of A*	48
3.3	Rough Terrain Adapted A*	49
3.3.1	Challenges with Higher Resolution Terrain	49
3.3.2	Increasing the Slope Threshold	50
3.3.3	Downsampling the Terrain	51
3.3.4	Increasing the A* Search Space	52
4	Performance and Results	55
4.1	Path Planning Performance in Rough Terrain	55
4.1.1	Speeding up A*	56
4.1.2	Time Complexity versus Optimality	56
4.2	BASALT Mars Analog Campaign Application	59
4.2.1	Overview	59
4.2.2	SEXTANT application use case during BASALT	60
4.2.3	Path planning through xGDS	61
4.3	Path Analysis	62
4.4	DEM sources	64
4.5	Results	66
4.5.1	Idaho 2016	66
4.5.2	Hawaii 2016 deployment	67
4.5.3	Hawaii 2017 deployment	68
4.6	Lessons learned	70
5	Path Planning Interfaces and EVA Navigation Aids	75
5.1	Introduction	75

5.2	SEXTANT Interfacing Architecture	76
5.2.1	Pextant API	77
5.3	Planning Interfaces	78
5.3.1	Past Interfaces	78
5.3.2	xGDS	79
5.3.3	SEXTANT Web Application	80
5.3.4	Geographical Information Systems(GIS) interface	81
5.4	Navigation Aids	81
5.4.1	Background	82
5.4.2	Direct Screen Display Based Navigation	84
5.4.3	Augmented Reality Display	86
5.4.4	Conclusion	105
6	Conclusion	107
6.1	Future work	107
A	SEXTANT Python API	109

List of Figures

1-1	Disadvantages of EVA planning by hand	20
2-1	Traverse planning datasets	28
2-2	General Planning Process	28
2-3	Mars Science Laboratory (Curiosity)	30
2-4	Path Planning Problem	31
3-1	EVA Path Planning Problem	38
3-2	Examples of obstacle maps	41
3-3	Walking speed from Apollo traverses, gathered and post-processed by Marquez (2007)	43
3-4	Santee et al. (2003) metabolic expenditure models applied to Apollo data by Marquez (2007)	45
3-5	Graphical explanation of A*	46
3-6	Slope Histogram	49
3-7	Challenges of rough terrain	50
3-8	Solution 1: Lowering the resolution of the dataset	51
3-9	Downsampling effects	52
3-10	Search kernels	53
4-1	Time complexity versus energy cost as a function of different heuristic factors and map downsampling resolutions	57
4-2	Comparing resolution and heuristic effects on path	58
4-3	Terrain encountered during BASALT deployment	60

4-4	Planning with SEXTANT in xGDS	61
4-5	Example of GPS processing of EV tracks	63
4-6	Overview of path planning map products used during BASALT campaign	65
4-7	Idaho 2016 plan for Mission Day 10 with high resolution terrain	66
4-8	Overview over Hawaii 2016 deployment, mission days that used SEX- TANT	68
4-9	Overview over Hawaii 2017 Kilauea Iki site, and plans produced by SEXTANT	69
4-10	Overview over Hawaii 2017 Keanakako'i site, and plans produced by SEXTANT	70
4-11	The importance of terrain	72
4-12	Velocity distribution	73
4-13	Planning with two different maps	74
5-1	High level overview of the SEXTANT API	77
5-2	Planning with SEXTANT in xGDS	80
5-3	Navigation during the Apollo days	82
5-4	Direct Screen Display integration during BASALT	85
5-5	SEXTANT WebApp design	86
5-6	Vision of Holo-SEXTANT	87
5-7	Hardware components used at Hawaii	88
5-8	Microsoft HoloLens Headset	89
5-9	HoloLens Reference Frame Transformations	93
5-10	First person view of Holo-SEXTANT UI	94
5-11	Holo-SEXTANT in use	97
5-12	Track from navigating with the HoloLens	98
5-13	Holo-SEXTANT first-person view	99

List of Tables

3.1	National Park Services constraints on slope	42
3.2	Table Type Styles	44
4.1	Dataset sources for SEXTANT during BASALT	64
4.2	Comparing planned and executed performance	70
4.3	Comparing planned and executed times	71
5.1	Previous traverse planning tools coupling computation and UI	79

Chapter 1

Introduction

1.1 Motivation

Future planetary missions to the Moon, and eventually Mars are predicted to last longer (NASA JSC, 2009; NASA, 2005) and cover larger distances than any previous planetary mission during the Apollo era. As already showcased by the Lunar Rover, and suggested in concept of operations for future planetary missions (Orz et al., 2013), these traverses will be executed in a mix of vehicular, and extravehicular activities (EVA) traverses. Although most of the distance would be covered by the vehicle, accessing rougher terrain - where one could expect to find interesting geology that a rover would not be able to access - would have to be done by foot (Chappell, 2004), and the distances covered might also be longer than those previously encountered.

Due to longer missions, and larger surface areas of science and exploration, the general activity of traverse planning will become significantly more important. As stated by Muehlberger (1981) "the name of the game when it comes to traverse planning is to maximize scientific output". As outlined in more details in Marquez (2007), the task of traverse planning encompasses deciding which targets to visit, what activities (instrumentation setup, experiment executions, sampling) to execute at each location, and the schedule: how much time is allocated for each activity, and which order it should be executed in. As the distances to cover between the stations increase, a subtask of traverse planning will become increasingly important: finding

efficient paths from one station to the next. This is the domain of path planning, and it consists in detailing the trajectory a vehicle or an astronaut will follow to traverse from one location to another. Given the fact that there is no trafficability infrastructure (paths, trails or roads) on the Moon or Mars, this is a critical phase of the traverse planning phase.

Efficient paths can be thought of as paths that both prevent the vehicle or the astronaut from going into treacherous terrain that could be of a safety concern, but also minimizes or maximizes a certain objective. This could be the amount of distance covered (or the total time one could get from one point to the next), in the context of the vehicles, which are generally limited to a maximum speed depending on terrain conditions (D-RATS). In the context of astronauts, this could be EVA resources such as oxygen, or CO₂ scrubbing capabilities, as these are finite resources that constrain the amount of hours that can be spent doing field work and traversing between different target points.

Path planning of planetary EVAs has historically been done manually: Apollo, our only example off the Earth, relied on a large team of supporting crew that would plan the traverses based on pen and paper maps, and physical three-dimensional topological maps, based on the planner's internal heuristics on what constituted a good plan, and how to best contour any obstacle and trying to make the traverse effort as simple as possible to the astronauts. In the years past Apollo, the closest we have gotten to similar conditions is through analog missions such as NASA's Desert Research and Technology Studies (DRATS) Orz et al. (2013), the domain of EVA path planning has not changed all that much: although the computer screen has replaced the pen and paper approach, and even the physical topological maps, the approach remains the same: that of human planning with internal heuristics based on the experience built up through the years.

The question of automation when considering planetary EVA planning was first asked by MarquezMarquez (2007). She evaluated the capability of manual path planning on a digital interface, and compared the planning performance of test subjects exposed to different map visualizations to the *optimal* plan calculated by an auto-

mated planning algorithm. The focus of the study, however, was on the performance of the human planning, and no assessment was done on the capabilities of the algorithm being used. Therefore, the automatically computed plan was taken as the ground truth, and there was no mention on limitations and the options that could have been considered to solve the automation problem, as the research was mainly focused on the human interface.

Further work was done by Lindqvist (2008) and Johnson et al. (2010a), and although both continued the development of the automated planning tool, their focus was mainly on implementing new features in the automated tool while improving the interface. The tool eventually got the name of SEXTANT: Surface Exploration Traverse Analysis and Navigation Tool. Essenburg (2008) did work on improving the automated algorithm, but yet there was little discussion on the actual performance of the planning algorithm and the limitations involved. Most use cases for this tool were also hypothetical lunar EVA plans based on proposed Lunar landing sites, and the work was very limited in its actual application in a real environment: the datasets were never larger than 500x500 pixels, the terrain encountered in the datasets was mostly smooth, and the resolution of the datasets was rather low.

These were all limitations that came into light when SEXTANT was applied in a real use case during NASA's BASALT (Biologic Analog Sciences Associated with Lava Terrains) research project, which was created with the goal to propose, implement and test new procedures and tools that would support future planetary missions (Deans et al., 2017). The first field deployment happened in June 2016 in Craters of the Moon National Park. The intentions were to use SEXTANT to do the traverse planning for the missions. However, SEXTANT relies on topological maps in digital forms: digital elevation maps or DEMs, and there weren't any readily available for the park. This was solved towards the end of the deployment, when a drone was flown over the mission area: it was used to generate both high resolution (3cm/pixel) imagery and DEMs. This came in great contrast to DEMs previously used for SEXTANT, with the highest resolution being 5m/pixel during Marquez work, but down to as low as 240m/pixel during Johnsons work. This showcased that the automation offered

by SEXTANT was not yet very robust, and motivated work towards understanding how SEXTANT could be adapted to deal with high resolution maps, and even ask the question as towards what the right resolution for such datasets should be. The larger datasets also pose the question on solver performance, a question which has seldom been raised in the past, other than at a qualitative level. This question ties back to the contingency question; the solver performance is rarely an issue during pre-planning phases, however it is much more critical during eventual contingency events, when time is at a premium. As outlined by Johnson et al. (2010a), "A future goal of SEXTANT is to be able to use it for real-time mission contingency planning in the field by the crew members".

Early on, BASALT also gave us insight into the importance of good planning, and how manual planning, even when given the right time resources does not always result in the best path. Poor plans can both endanger, and delay significantly the mission. This is illustrated in Figure 1-1:

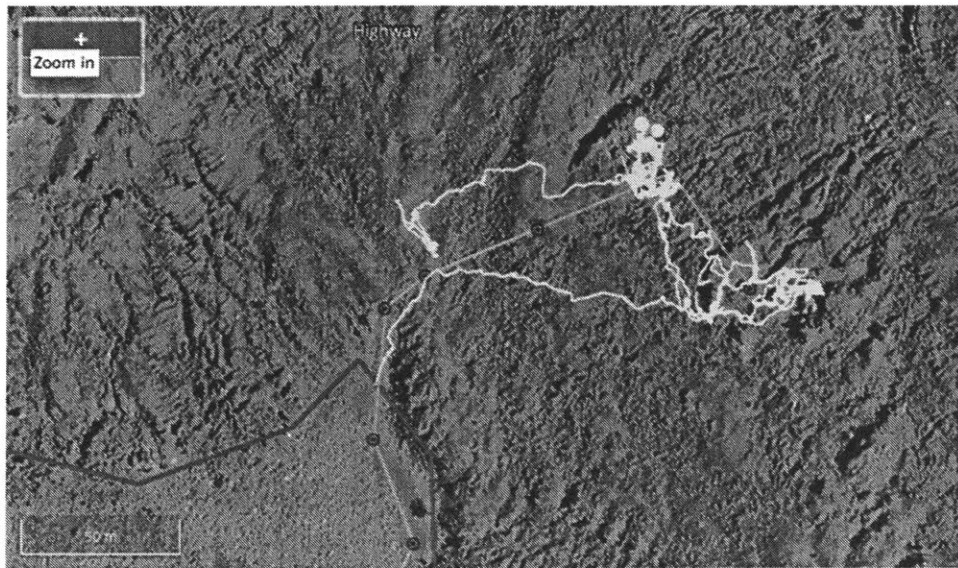


Figure 1-1: Comparison of manually planned EVA (orange), and executed EVA (green)

As seen in Figure 1-1, the EV Crew had to walk far off the planned path at several iterations, this occurred because the EV Crew realized in the middle of the simulation that the path they were following was taking them into two steep terrain that they

could not walk through.

Granted, the human planners only had a very low-resolution image to generate the plan, and did not originally rely on any elevation data, yet this examples goes to show important it is to not only have a good plan, but also to use the right data sets to do the planning. Although planning based on imagery can to some extent work, in challenging terrain imagery only will probably not be enough.

On a later occasion, high resolution data sets for the terrain where actually available, and although the plan that came out based on this data was much easier to traverse, it did take a significant amount of human planning time to come up with the traverse. This would be a limiting factor during an eventual contingency or tactical decision change made mid-mission, as the challenges of quick decisions under time pressure might send the astronauts on a non-ideal path, given. As we start to think of more distant destinations, such as Mars NASA JSC (2009) re-planning manually from Earth becomes virtually impossible, due to the significant communication delays.

1.2 Thesis Contribution

1.2.1 A Formalization of Automated EVA Traverse Planning

This thesis formalizes the general traverse planning problem as a cost minimization problem on a Euclidean surface, and discusses a wider set of algorithms that have been covered in the literature in the fields of robotics, game development, and computer science in general. It still ties in the limitations that make EVA traverse planning different from other applications, and especially goes into discussing the limitations on the different datasets used in these different algorithms.

1.2.2 Performance Evaluation of an EVA Planning Algorithm

General performance metrics such as time complexity and space complexity for the A* algorithm used are derived from the theory; in parallel different case studies were run and times were computed depending on different planning options that could increase

or decrease the fidelity level and the optimality of the path. The cases span a mixture of hypothetical edge case scenarios, to illustrate the limitations of the algorithm, and real plans used in the context of BASALT.

1.2.3 Automated Traverse Planning Limitations

Although previous works by Marquez (2007) have discussed limitations of automation when it comes to its use in the context of a *Decision Support Aid* tool, this thesis makes the case for additional data beyond an altitude map in the use of automated traverse planning, specifically information on terrain properties with respect to traversability and motivating the determination of human traversability performance in different terrain types. It also outlines how such data could be used with the current planning algorithm, or how it could be used in the context of a different planner.

1.2.4 Experience from an Analog Use Case Environment

Automated Traverse Planning had been applied previously in terrain which offered very limited challenges, and where many unwritten assumptions on the conditions for the planner to work happened to check. Yet when confronted with a challenging analog environment such as that encountered during BASALT, many of these assumptions surfaced, pushing for a large set of changes to improve the automation capabilities. This thesis outlines many of these experiences, and is the first full use case of Automated Traverse Planning as a critical capability during and analog EVA execution.

1.2.5 An Open Source, Free, User Friendly Code Base

Previous implementations of SEXTANT was first done in Java, which is prominently less used by engineers and research scientist, and was later ported over to an licensed environment: MATLAB, more commonly used by this user community. The code has now been ported to Python, widely used by the same user community. It also makes

the installation of SEXTANT out of the box, requiring no licensed software for it to run. Finally all code is available on an online repository: github.com, which is widely used across the world for hosting open source projects. The repository is also under version control, which allow current and future contributors to track the changes in the development environment.

1.2.6 A Flexible Interface and Integration with Several Tools

This thesis outlines the SEXTANT Application Programming Interface(API) details, which allow other programs and tools to use the SEXTANT capabilities as a black box. This interface has been tested for three different use cases: first, the Exploration Ground Systems(xGDS) tool developed by NASA Ames and used for planning the traverse plans in the context of BASALT. Second, the Sextant WebApp, which was developed originally as a standalone offline interface for SEXTANT usage in the field, and then eventually was enhanced by a collaboration with the NASA AMES xGDS team and a team at Cornell University to become the navigation tool used by the EV crew members during their traverses. Third: the Holo-SEXTANT application, an Augmented Reality heads-up display based on Microsoft's HoloLens, which was developed and explored as a navigation aid, and tested during the second BASALT deployment.

1.3 Thesis Outline

This thesis is organized as follows: in Chapter 2 it discusses the background work relevant to this thesis, in Chapter 3 it goes into the formalization of the path planning problem, and the implementation of the solver, in Chapter 4 we discuss performance of the planner, and its application to BASALT, in Chapter 5 we discuss it's integration with other systems, and it's use during two field deployments. Chapter 6 brings back the big picture, and outlines future research directions that span from the current work.

Chapter 2 outlines some history on the activity of traverse planning as it has been

executed in the past through the Apollo missions and current analogs, the background of SEXTANT from its inception, and argues the case for an explicit separation between the user interface and the automated traverse planner, as previously these two components had been considered together. It also explores the general subfield of path planning in computer science, outlines several of the general algorithms that could have been applied, and independent applications of such algorithms where it worked successfully, and takes account for the considerations to be made with each of them.

Chapter 3 poses the generalized formalization of traverse planning as the minimization of an accumulated cost function over a surface. The cost function investigated in the largest extent is that of energy consumption associated with the terrain traverse, as one could expect this to be highly correlated to consumption of life support resources, which are finite during EVAs, and would be a function in great interest of minimization. Other cost functions, such as distance, traversal time and others which have been explored in previous work are also mentioned to some shorter extent. This chapter also outlines the current planning algorithm based on A*, and how this is one potential method to solving the optimization problem.

Chapter 4 contains a large set of major contributions to the thesis. It outlines the performance of the planning algorithm in terms of time complexity, both from a theoretical standpoint, and from running the algorithm on several test cases. The chapter then goes into some speed-up strategies to increase the performance: implementing the traverse algorithm with vector type operations, and putting a weighting factor on the heuristic function from the A* algorithm, which has been mentioned in the literature several times. The the chapter covers results from applying SEXTANT during the BASALT analog deployment.

Chapter 5 wraps up with the bigger picture, how to interface the path planning algorithm into the planning pipeline. We describe the Automated Programming Interface (API) developed to allow other users to use our implementation, and link to the code repository which is easily accessible on-line. This chapter also dwells into a look on path planning user interfaces and the human navigation problem, specifically

astronaut oriented. Path planning user interfaces could be used in preparation for the mission, but also during the mission: we present both. The chapter then goes more into detail on how to facilitate navigation of the path through two different interfaces: a digital display based interface, and a heads-up augmented reality display. Although no rigorous user testing was completed, qualitative feedback from crew testing in a simulated environment is presented.

Chapter 6 concludes and highlights the key contributions of the work presented in this thesis and expands upon areas of future research for SEXTANT, both on the implementation of new algorithms, and on how to continue the development of navigation tools, and suggestions for different interfaces to be tested in a more rigorous manner.

Chapter 2

Background

This section first introduces the broader idea of EVA planning, and then scopes it down to the specific problem that SEXTANT addresses: the path planning problem. Finally this section gives some more background on the BASALT campaign that drove the development of SEXTANTs new features.

2.1 Context of EVA Planning

Many of the decisions made during traverse planning rely on map products and other datasets to inform these decisions. This includes, but is not limited to photographic maps, multi-spectral imagery, topological and geological maps, estimated times for the set of activities to be executed, and relevance of these activities to the bigger mission goals. Path planning has historically relied on two main datasets: regular imagery and topological maps of the terrain altitude. Topological maps, which when digitized are known by the name of Digital Elevation Maps, can be further processed to produce slope maps and hillshade maps, which can then further be informative to the planning effort.

As shown in Figure 2-1, several of the map datasets are used for different types of planning: the science team uses multispectral, visible, and even thermal imagery to determine the composition of the ground, and generate different processed data products, such as geological map, which highlight areas of high scientific interest,

and serve for the selection of candidate science sections. The activity planning can be informed from the slope map, or a geological map, to determine areas that are safe for different experimental setups, or for the crew to be there in the first place. Finally, as previously mentioned, path planning focuses on topologically derived data, and visible imagery, as highlighted in Figure 2-1.

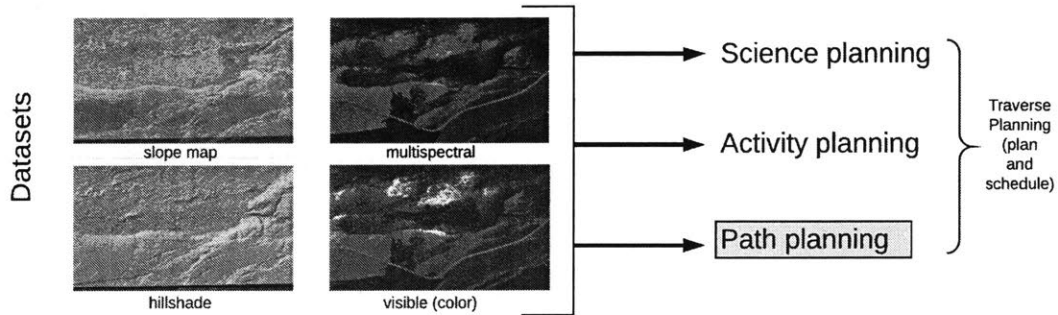


Figure 2-1: Datasets used in traverse planning

Although the discussion on datasets requirements, and resolution requirements, is important for all parts of the traverse planning activity, this paper focuses solely on these questions as they pertain to the path planning problem.

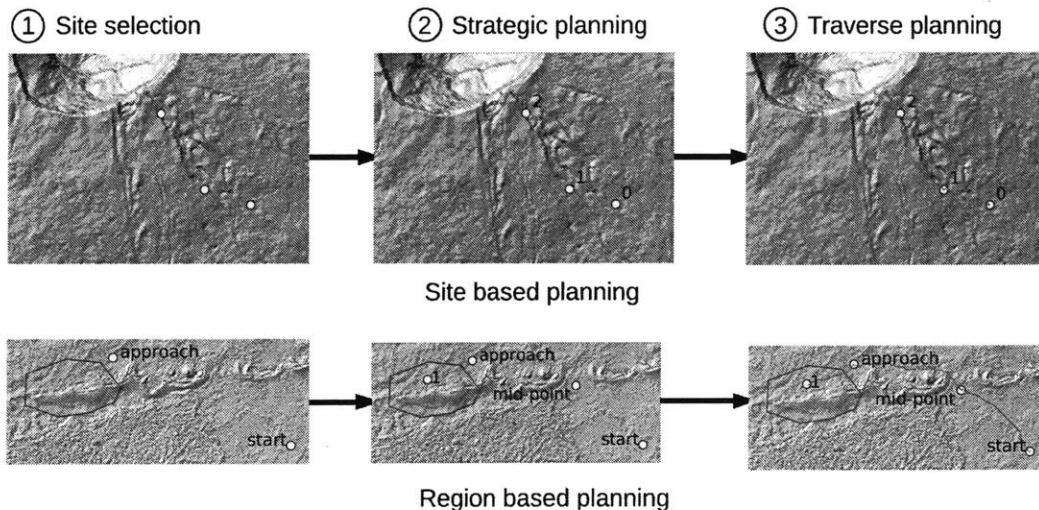


Figure 2-2: General Planning Process

Furthermore, the focus of path planning is low level: the goal is to determine the best trajectory to follow between several stations. This stands in contrast to

deciding the order in which certain stations will be visited, or determining the type of concept of operations, for example re-visiting stations on the path back to base, or only visit stations once and do a loop. (probably need some input from exploration here). Figure 2-2 illustrates how path planning fits in the general planning pipeline in two different operational concepts evaluated during the BASALT campaigns: a station based approach, which was used during the Apollo missions, and for previous analog missions such as D-RATS and AMADEE-15.

2.2 The Path Planning Problem

2.2.1 Areas of application

As described by Johnson, the path planning procedure for each Apollo lunar mission was manual and based on simple photomosaic and topographic base maps (Muehlberger, 1981) produced by previously manned and unmanned missions. Due to the uncertainty in the map products, there was already the awareness that the astronauts might encounter obstacles, and that the plan would take longer than planned. As a result, the EVA crew had to constantly be prepared to change their route as they better discovered the real conditions of the environment they were traversing. For any replanning they had to wait for instructions from the planners and engineers on Earth, slowing down the mission even further.

A similar problem exists in the world of robotic planetary missions, where rovers must find an obstacle free path in their environment. However, here they must rely on autonomy to generate and execute the plans, as communication delays make teleoperation intractable, and plans hard to change if operators discover challenges in the environment. Autonomous methods have been applied to the Mars Exploration Rovers: Spirit and Opportunity, and the Mars Science Laboratory(Curiosity) (Bajracharya et al., 2008). Path planning is also an important problem in the wider field of robotics and video games. In the realm of video games, it is used to generate paths for non-player characters(NPCs), or to aid the playing character reach a goal

within the game(Abd Algfoor et al., 2015). It has also been designed to automatically generate generic roads to build naturally looking virtual environments(Galin et al., 2010). In robotics, it is applied to autonomous vehicles, such as the DARPA Urban Challenge in 2007(Urmson et al., 2007), robotic locomotion, applied in the DARPA Robotics Challenge(Deits and Tedrake, 2015), or unmanned aerial vehicles(UAVs). A recent paper also applied it to an unmanned water surface vehicle (Shah, Brual C, 2016).

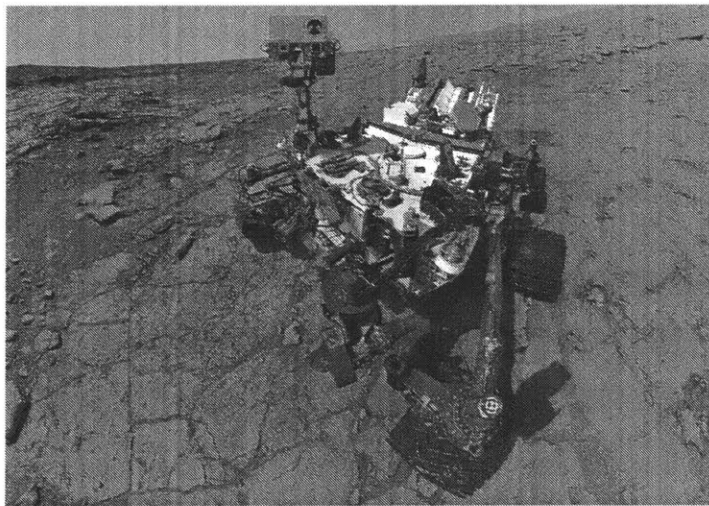


Figure 2-3: Mars Science Laboratory (Curiosity)

2.2.2 Problem definition

The most general path planning problem consists in finding an unblocked path in a given environment from a start location to a goal or target location, while minimizing the cost of traversing the given path(Daniel et al., 2010). From this point of view path planning can be thought of as an optimization problem, where the design variables are the coordinates of the path, and the objective function is the cost. The constraints are then related to the obstacles in the terrain, and other restrictions on how the vehicle can move in its surroundings (such as vehicle dynamics). When the objective function, or the cost, is simply the distance traversed, the path planning problem is also known as the shortest path problem. In many robotic problems the shortest path

problem is constrained to be on a surface; this problem is also known as the geodesic problem (Mitchell and Papadimitriou, 1991).

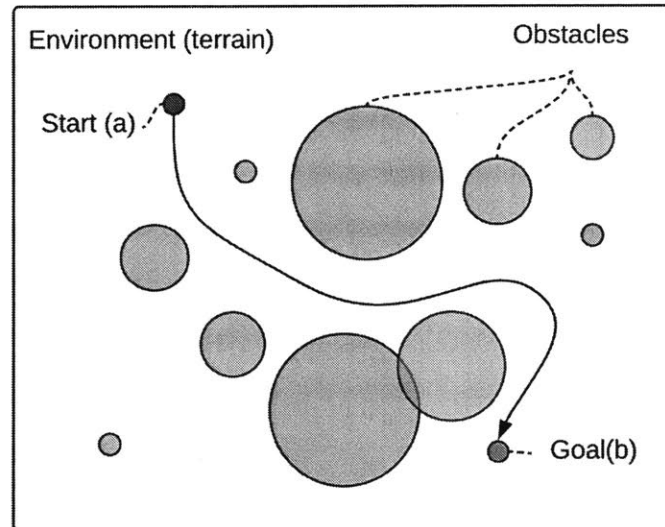


Figure 2-4: Path Planning Problem

2.2.3 Problem Representations

A large set of algorithms have been developed to solve the path planning problem. Here we limit the description to the algorithms that constrain the path to a surface or terrain. The algorithms are tightly connected to the environmental representation and can be separated into two categories: grid-based techniques, and hierarchical techniques (Abd Alfoor et al., 2015). Grid-based techniques rely on transforming the environmental representation into a weighted graph, which abstracts the problem to a level where more general search algorithms can be applied. In the weighted graph representation, nodes in the graph correspond to locations, and edges represent actions that an agent can take to move from one location (node) to the next, while the weights represent the cost associated with a specific action. This allows for more general graph search algorithms to be applied. Grid-based representation can further be split into regular grid representations, and irregular grid representations.

In regular grid representation, nodes are connected to nodes representing neighboring cells in the grid representation, for example in a 2D square-grid, a node rep-

representing each cell is connected to its eight neighboring cells (octile representation). This is by far the most common environmental representation, as it is also very simple. Other representations use a 2D hexagonal-grid, nodes are connected to six neighboring nodes (hexagonal representation). The regular grid representation is a natural representation for many of the environments encountered in the motivating application areas: the grid provides a natural representation for obstacles, for example through a binary matrix, where ones represent the presence of an obstacle in a certain location. Grids are also the standard format to represent digital elevations maps (DEMs). Another advantage of grids is that the performance of any algorithm relying on its structure could in theory be improved by increasing the resolution of the data.

Several irregular grid representations exist, but most applied are triangular meshes and visibility graphs, the latter which is limited to 2D environments. A visibility graph relies on the fact that shortest paths in two-dimensional environments constrained by polygonal obstacles follow straight lines except at the vertices of obstacles, where it might turn. Therefore, the visibility graph consists of obstacle vertices, where the actions, or edges of the graph at each node consist of which vertices are visible from the current location. Graph search can then be applied to this representation. Although irregular triangular meshes can also be abstracted as graphs, on which search can be applied, it can very often be suboptimal. Mitchell and Papadimitriou (Mitchell and Papadimitriou, 1991) developed a shortest path algorithm that is optimal within to a specified tolerance. Aleksandrov, Maheshwari, and Sack (Aleksandrov et al., 2005) developed a generalized path finding algorithm for weighted triangular representations of the terrain, which in practicality means that it can capture the path planning problem when the cost function is more complex than the distance. However, both methods relying on the triangular representation have limited applications in an obstacle prone environment (Ferguson and Stentz, 2006). As Ferguson wrote concerning methods developed on top of Mitchell, Mound and Papadimitriou's work: "These approaches are efficient for planning through environments containing a small number of homogenous-cost regions but are computationally ex-

pensive when the number of such regions is very large, as in the case of a uniform grid with varying cell costs”

2.2.4 Path Planning Algorithms

Since many environment representations can be abstracted as graphs, many path planning algorithms rely on generalized graph search algorithms. The most widespread graph search algorithm is Dijkstra. When applied to a graph it builds up a path (a sequence of nodes) from the start node towards other nodes on the graph while always keeping track of all paths of equal length. This guarantees that when the algorithm reaches the goal node, it will be the shortest path. Hart developed a graph search algorithm that improves the speed of the algorithm by only searching for paths that are in the general direction of the goal. This type of search is called heuristic search, and the algorithm is called A*(reads a-star). This algorithm is very general and can also be applied to a much larger set of problems beyond path finding.

Extensions of A* have been developed in the context of path planning. One problem encountered in robotics is that the representation of the terrain is often of low quality at the time of planning, and details of the terrain are not discovered until the autonomous vehicle can sense the environment in greater detail. Dynamic A*, or D* for short(Koenig and Likhachev, 2002) addresses this problem updating the relevant nodes and edges of the graph and letting the changes propagate through the graph. As the environment and its obstacles are more precisely mapped, D* Lite can speed up replanning after an initial search by up to two orders of magnitude. The octile grid representation of the environment is very frequently used, however it can be up to 8% suboptimal due to the restriction that a path can only consist of a set of straight and diagonal paths(Ferguson and Stentz, 2006; Daniel et al., 2010). Ferguson and Stentz developed the Field D* algorithm while still relying on a gridded representation of the terrain; although instead of only allowing straight line connections between cell centers in the octile representation, it tracks cell edges, and allow lines to cross the cell without having to go through the center, thus creating better, and more realistically looking paths. Daniel et al. developed the Theta* algorithm, which considers straight

line paths to cells beyond the immediate neighborhood of a cell, if such paths do not intersect any obstacles. Daniel et al. compare optimality of A*, Field D* and Theta* showing how Theta* outperforms both, at a slight increase in computation time.

In addition to grid-based methods, path planning can also be solved through hierarchical techniques; the goal here is to lower the amount of memory needed typically to store larger grids and speed up the solution time significantly by reducing the search space while giving up on optimality. This family of methods include Rapidly-Exploring Random Trees(RRT)(LaValle, 1998), and RRT*(Karaman and Frazzoli, 2011). It also includes searching on quadtree representations of the environment. Algorithms have also been developed on top of the existing algorithms to speed up the solution time by using heuristic and techniques that no longer guarantee optimality; this includes Anytime D*(Likhachev et al., 2005) and Anytime RRT*(Karaman et al., 2011)

2.3 SEXTANT background

An EVA traverse planning tool was originally developed by Carr(Carr et al., 2003), under the name of *The Geologic Traverse Planner*. This tool allowed a human operator to design and analyze traverse paths by hand while outputting key metrics such as metabolic cost of transport and visibility of landmarks and points of interest. It would also check for compliance with mission rules on maximum slope, maximum energy, total EVA time and EVA duration, and notify the user of any violation. Carr also defined a metric that was named the sun cost, which assessed the sun position along the trajectory of the path. Lower sun scores are desired and obtained from trajectories perpendicular to the direction of the sun, as astronauts had reported contrast and depth perception problems in directions of up- or down-sun. The Geologic Traverse Planner was applied to a case study on the second EVA during the Apollo 14 mission to Cone Crater on the Moon, and the tool was used to improve the plan based on the analysis generated.

Marquez first introduced the notion of automatically generating the path, as

opposed to having an operator manually designing them, in the Planetary Aid for Traversing Humans (PATH) tool (Marquez, 2007). This was used as a research platform to evaluate the benefit of different interfaces to aid manual traverse planning and evaluate the effect of different levels of automation. Test subjects were asked to generate plans with different level of automation and different visualizations, and the performance of the plans was assessed with several of Carr's metrics. While it is easy for a person to generate plans that avoid obstacles with the right map, it is less intuitive to optimize a cost function without the right visualizations. Marquez also introduced by Marquez: the exploration cost, which combined metabolic cost with the sun score. Her research was the first to formalize the EVA path planning problem in the context of automation. Path planning was accomplished with Dijkstra's algorithm, modified to minimize the exploration score instead of distance, which is the conventional cost function. Wood and Wood (Wood and Wood, 2006) had applied a similar approach for energy cost, but to a very different problem. In their study they made a note on the significant computational resources required to run this algorithm. Marquez and Wood and Wood both used the algorithm to generate metabolic energy (caloric) cost heat plots from a specific plot, which Marquez proved was a very intuitive user aid to generate visually aided manual plans, as the shortest path on such a plot can easily be found from a following a steepest descent gradient, which speeds up the time it takes for a person to find an optimal path. Marquez applied PATH to a slightly modified case study from Carr, where she introduced artificial craters and hills in the Apollo 14 terrain to make the planning process more interesting during user testing.

Johnson B. was the first to change the solver from Dijkstra's algorithm to A*. In practice this resulted in faster solution times, improving the performance of the planner. Work focused on the interface was done by Lindqvist, who developed a more user-friendly interface connecting PATH with ArcGIS, a geographic information system (GIS) software, that easily integrates visualization of the terrain and map products (Lindqvist, 2008). Essenburg implemented a combined planning and visualization tool in MATLAB that was called Pathmaster (Essenburg, 2008), which

also allowed to generate plans for multiple astronauts in the same environment. Es-
senburg was the first to apply the tool to a terrestrial testing environment, performing
qualitative functional testing at the Jet Propulsion Laboratory's Mars yard for simple
scenarios. Johnson expanded SEXTANT to perform path planning for any surface
explorer including astronauts, vehicles and autonomous rovers. Johnson added cost
functions for robots, included new models for thermal performance and considered
shadowing effects on the Moon, a key factor during lunar traverses on polar latitudes.
Gilkey et al. performed the first quantitative testing of SEXTANT during a field
study conducted at the Mars Desert Research Station (MDRS) in Utah (Gilkey et al.,
2011). Results showed significant under-predictions for time estimates and energy
cost.

Chapter 3

Path Planning in Rough Terrain

This section gives an overview of SEXTANTs path planning back-end, the changes that had to be made to adapt it to ensure successful for its use in a real case, and how it was used by the traverse planning team. Although faster path planning algorithms exist as outlined in Chapter 2, SEXTANT still relies on the A* algorithm on an octal grid despite the drawbacks associated with sub-optimal straight and diagonal path sections. The underlying reason is the simplicity to implement A* in any coding language and its flexibility to be used for more advanced formulations. This justifies its use in other applications such as proposed path planning for future polar prospectors (Speyerer et al., 2016).

3.1 Path Planning Problem Formulation

As outlined in Chapter 2, the path planning problem consists in finding an unblocked path in a given environment from a start location to a goal or target location, while minimizing the cost of traversing the given path. In the context of EVA traverse planning, the path is constrained to lie on the terrain and must avoid all obstacles present in the terrain. The key decisions in the formulation is describing the environment, captured by the octal grid, describing obstacles, determined based on slope angle thresholds, and describing the cost function, which for BASALT was based on a human metabolic performance model. Finally, SEXTANT introduces the use of

waypoints between start and goal to give the planner more control of the exact direction of the path. These elements are illustrated in Figure 3-1, where we recognize the elements from Figure 2-4 from Chapter 2. These elements are next discussed in greater detail.

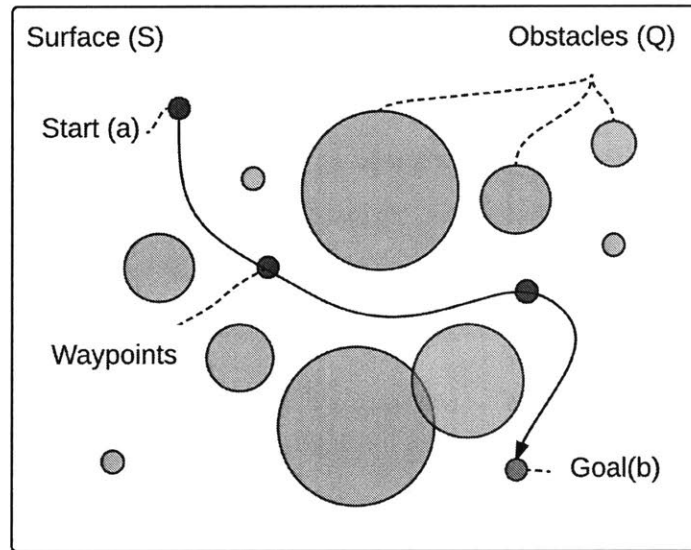


Figure 3-1: EVA Path Planning Problem

3.1.1 Waypoints

One of the findings during studies comparing automatic path planning, assisted path planning and fully manual path planning (Marquez, 2007), was that allowing the user to constrain the traverse by forcing it to go through a specific point would empower the user. This is just an extension of the path planning problem, where the waypoints become targets of previous intermediate path and starting points for the next intermediate path. This is not only helpful from the point of view that a user can override any output from the SEXTANT algorithm, but also allows users to provide a sequence of points that could match waypoints that need to be visited, such as science stations, during a longer traverse. During the BASALT 2016 campaign, the start was set to the beginning of the approach, and waypoints were established for up to three science stations within proximity of one another. Waypoints were only used

on one occasion to redirect the plan generated by SEXTANT. During the BASALT 2017 campaign, SEXTANT was only used to plan the path up to a larger region of interest, due to a change in concept of operations.

3.1.2 Environmental Description of the Terrain

To capture all the data related to the terrain we will use the term Environmental Model. Since it's difficult to store the surface representations in a continuous model, we will need a discretized model that will be part of the environmental model. Topological data is generally available through two formats: a digital elevation map(DEM), which often go by the name of Digital Terrain Maps(DTM) or Digital Surface Maps(DSM), or a triangular representation such as an Irregular Triangulated Network (TIN). DEMs have a similar data structure to digital images, where each pixel represents an elevation; this part of the DEM is often denominated a height map. Additionally, DEMs contain information on the resolution, i.e. the distance between two contiguous pixels in the height map. For geographically tagged DEMs, the data set also includes the location of the terrain. TINs are conventionally represented by a set of vertices (a point cloud), and a list of triangle faces, where each triangle is identified by the number of the vertex points which form it. TINs can be generated from LIDAR data through Delauney triangulation of the points.

Terrain data can come from several types of sensors: Light Detection and Ranging (LIDAR) sensors, infrared sensors, or optical imagery which can be post processed if there are at least two pictures with different viewpoints of the same area. Such sensors can be deployed on satellite, manned and unmanned aircraft and ground-based surveys. Depending on the sensor, and the method used, the resolution, accuracy, and size of the dataset will vary. However, as sensors have become cheaper and more capable, today we can gather higher resolution, more accurate, and much larger datasets which would not have been available as a source for traverse planning earlier on.

Often the only available datasets on general areas is satellite data: this is especially true when airborne surveys might be limited. Satellite generated DEMs generally

come at a low resolution: state of the art on the commercial satellite market offer 12 by 12m data sets. However, satellite imagery is readily available and come down to 30cm resolution(Shan et al., 2016). DEMs can then be developed from a technique known as stereo photography: NASA Ames has developed the Ames Stereo Pipeline(Shan et al., 2016) which triangulates the terrain based on two images of the same area taken at slightly different times. This algorithm can in theory reconstruct DEMs at the same resolution as the source imagery, however it is recommended that this be down sampled to four times the resolution in order to prevent noise created by the algorithm. For Earth satellite imagery this translates to a 1.2m resolution imagery out of the box. This is a good backup technique and insured we had terrain maps for certain areas that were not covered by airborne LIDAR.

3.1.3 Obstacles

Obstacles are areas on the surface that cannot be traversed by the path. This could be *object based* obstacles, such as trees or boulders, or properties of the terrain that make it inaccessible or dangerous, such as slippery terrain(ice). SEXTANT focuses on one type of obstacle specifically: steep terrain, i.e avoiding crevasses, walls, or sudden depressions in the terrain. Although obstacles can be entered manually, SEXTANT automatically computes an obstacle map based on the terrain representation. Several methods can be used to calculate the slope depending on the representation of the terrain. For a DEM, slope can easily be calculated from a second order accurate central difference method, where the gradient with respect to an axis, for example x is described by equation 3.1:

$$f'_i = \frac{f(x_{i+1}) - f(x_{i-1}))}{2\Delta r} \quad (3.1)$$

Here f'_i is the value of the gradient at a specific cell, and $f(x_{i+1})$ and $f(x_{i-1})$ are the altitudes along a certain axis(\hat{x}) before and after the current cell. Δr represents the cell spacing, i.e. the map resolutions. The slope is then calculated according to 3.2:

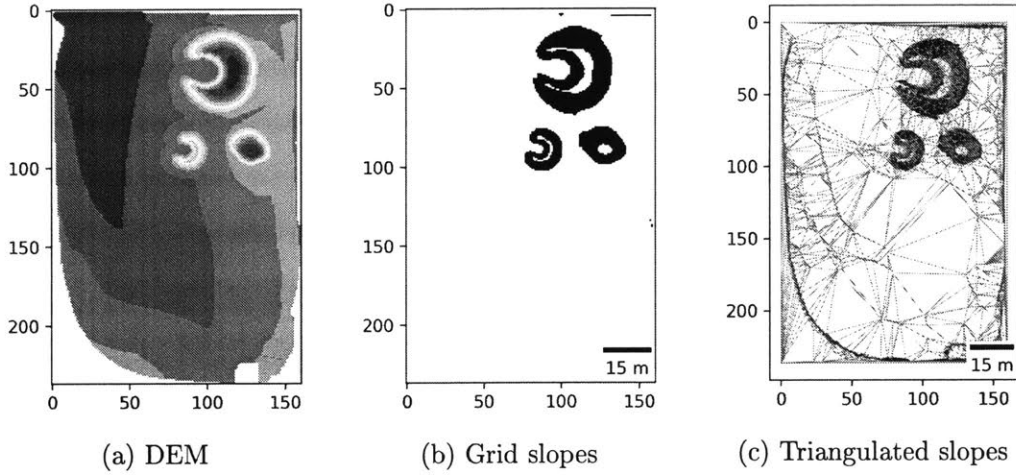


Figure 3-2: Obstacle map for example terrain, here, NASA Ames Roverscape with 0.5m resolution. Areas in black have a local gradient larger than 10 degrees.

$$\theta(x, y) = \arctan \left(\sqrt{f(x)'^2 + f(y)'^2} \right) \quad (3.2)$$

If the terrain is represented by a TIN, or other polygon, slopes can be calculated from the normal faces of the polygons:

$$\theta_k = \frac{\hat{\mathbf{n}} \cdot \hat{\mathbf{i}}}{\|\hat{\mathbf{n}}\| \cdot \|\hat{\mathbf{i}}\|} \quad (3.3)$$

Where $\hat{\mathbf{n}}$ is the normal unit vector of a given polygon, and $\hat{\mathbf{i}}$ is a unit vector along an horizontal direction. Figure 3-2 shows this:

The slopes are calculated along the direction of steepest descent on a grid point or polygon, depending on the representation. Although one could be traveling along the direction perpendicular to the steepest descent, i.e. along a contour of zero slope, in reality any agent traversing in such a perpendicular fashion would have a geometry that might still make it challenging due to a length scale in the direction the steepest slope.

National Parks Service (1996) have outlined a set of standards for construction of segments of *semiprimitive trails*, displayed in Table 3.1. Although walking in an pressurized suit will limit the ability to traverse terrain that is already challenging on

Earth, some of the limits imposed by the NPS can be used as a guideline for slope thresholds.

Table 3.1: National Park Services constraints on slope

Property	Slope threshold
Maximum slope	30%(16.5°) for 30m
Sustained slope	15%(8.5°)
Cross-slope (perpendicular to direction of travel)	8%(4.5°)

Table 3.1 also references the maximum cross slope, which further justifies the statement that although one could travel perpendicularly to the direction of the terrain for zero slope along direction of travel, one would still be limited by the cross-slope limit.

3.1.4 Path Planning Metrics

Carr et al. (2003) outlined several metrics of interest in the analysis of an EVA path. The simplest metric is the distance, and is independent of the agent executing the path. The list also includes traverse time, visibility of landmarks along the path, relative sun angle to the direction of travel, and energy consumption. Marquez (2007) aggregated the sun angle and energy consumption metrics into an *Exploration Score*. Finally Johnson et al. (2010a) included the heating power demand for both astronauts and rovers and the cooling demand from sublimating water as important metrics to consider. The metrics can be used in the path planning effort as cost functions that aggregate over the traverse path, and that we want to minimize. The current version of SEXTANT has left out many of the previous metrics and capabilities such as thermal management due to the application of SEXTANT to terrestrial analogs. Right now SEXTANT incorporates the distance, the total traverse time, and the metabolic cost function, for astronauts and not rovers. These are discussed in more details below:

Distance Metric

This distance metric is the most common cost function in path planning. This problem is known as the shortest path problem, and many of the techniques outlined in Chapter 2 are adapted out of the box to solve this problem, such as the algorithm described by Mitchell and Papadimitriou (1991), which is specifically for triangulated terrains.

Total Traverse Time

If the velocity of the explorer was considered constant with regards to the terrain, this could be transformed into an equivalent problem to the shortest distance problem. However, both for humans and for rovers there is generally an uneven velocity distribution which is highly correlated to different terrain factors, one very important one is the slope. Since the automated path planning that lead to SEXTANT was originally developed within a Lunar context, Marquez (2007) extrapolated velocity data from the Apollo traverses shown in Figure 3-3. These velocities were considered parallel to the surface.

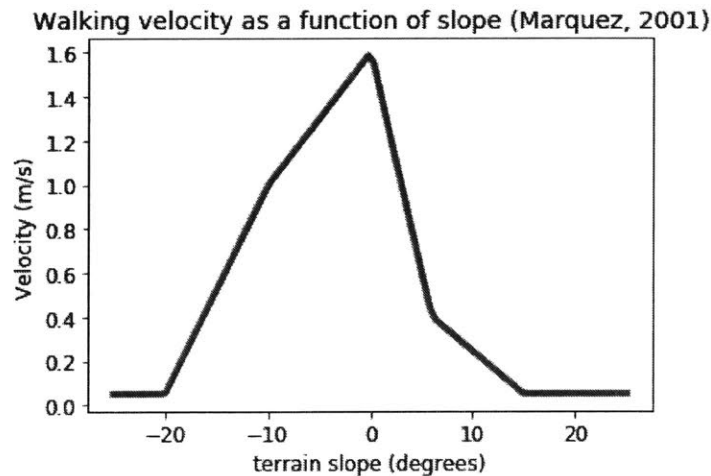


Figure 3-3: Walking speed from Apollo traverses, gathered and post-processed by Marquez (2007)

As expected, an astronaut would move slower both up-hill and down-hill. The model is currently only defined between negative -20 degrees and 15 degrees of slope,

and is held constant at some lower value outside these bounds which will normally either be considered obstacles, so not traversed at all, or heavily penalized in any path due to the slow traversal time.

Metabolic Consumption

This metric estimates the power required by the astronaut to walk in certain terrain. One could hypothesize that this metric is highly coupled with consumables in an astronaut suit environment, such as oxygen and carbon dioxide, and therefore the cumulative power(the energy) can be used as a proxy for consumables. The curve depends both on the walking speed(which is already a function of the slope), and the slope, giving us the relationship in Table 3.2 from studies done by Santee et al. (2001).

Table 3.2: Table Type Styles

Metabolic Rate(W) = $W_{level} + W_{slope}$	
$W_{level} = (3.28m + 71.1) \cdot 0.661v \cdot \cos(\alpha) + 0.115$	
Slope α	W_{slope}
$\alpha = 0$	0
$\alpha \geq 0$	$3.5 \cdot m \cdot g \cdot v \cdot \sin(\alpha)$
$\alpha \leq 0$	$2.4 \cdot m \cdot g \cdot v \cdot \sin(\alpha) \cdot 0.3^{\frac{ \alpha }{7.65}}$

Where m is the total mass of the person and the suit or any other equipment, g is the acceleration due to gravity($9.8m/s^2$ on Earth), α is the slope of the terrain, and v the velocity parallel to the terrain, making $v \cdot \sin(\alpha)$ the projected vertical velocity, and $v \cdot \cos(\alpha)$ the projected horizontal velocity.

The model has two parts: level walking W_{level} which depends only on the horizontal velocity, and W_{slope} which depends on the slope. Santee's original study was only evaluated at the velocity of 1.34 m/s, and for slopes in the range of -12 to +12 degrees. The model was extended to allow for other velocities based on models by Passmore and Durnin (1955), given the model presented. Santee's load carriage model was then combined with Marquez's velocity model to result in a metabolic

consumption rate model as a function of the slope of the terrain. Figure 3-4 shows metabolic consumption rates for different crew masses in a terrestrial gravity environment. The flattening of the curves under -20 degrees and above 15 degrees are due to the velocity model by Marquez as discussed in the previous subsection. As can be seen, in shallower terrain, the model is not too sensitive to the mass, but in steeper areas, especially uphill, the differences become significant. This suggests that during the path planning effort, heavier astronaut crew will generate paths that trade walking longer and shallower over short and steep.

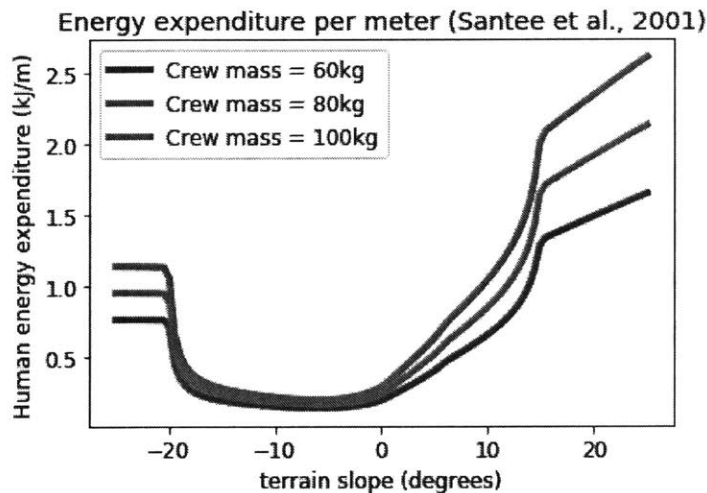


Figure 3-4: Santee et al. (2003) metabolic expenditure models applied to Apollo data by Marquez (2007)

3.2 Path Planning Algorithms

3.2.1 Original A*

Since implemented by Johnson (2008), SEXTANT solver has been built upon the A* algorithm. Although introduced in Chapter 2, here is a more detailed description of how A* is applied. Given a start point, and a goal point, which consist of two waypoints, the algorithm starts by considering the eight neighboring cells of the start point, excluding obstacles, as candidate points for the shortest path. This process

will also be referred to as expanding a given cell. Each cell is associated with a coordinate and an elevation. The next step in the algorithm estimates the cost to get from to current point (for now the start) all the way to the goal when passing through a candidate cell. This cost is the sum of the cost of moving from the current cell to a neighboring one (for example cell 3, as shown in Figure 3-5) and an estimate of the cost to get from the neighboring cell to the goal. The estimate is generated by a heuristic function, further detailed in next section. Each neighboring cell is associated with the estimated cost, and the cell with the lowest cost (e.g. cell 3) is selected. The procedure is repeated with the selected cell. The cell is expanded and each of the neighboring cells get their cost assigned. This time, in addition to excluding obstacles, previously expanded cells are also excluded from the candidate neighbors (e.g.: the start cell would be excluded). The final detail is that if a cell has previously been considered as a neighbor of another cell in the search so far (e.g. Cell 2 was expanded by the start cell), and thus assigned a cost, the cost associated with this cell is only updated if it is smaller for the new cell.

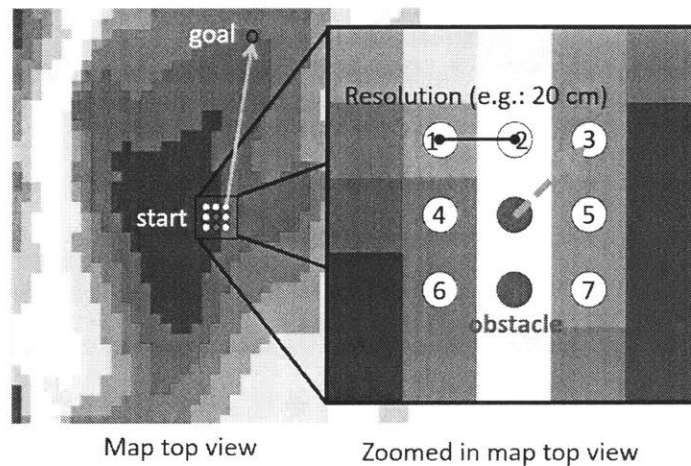


Figure 3-5: Graphical explanation of A*
 Similar colors represented locations of roughly similar altitudes

A great tutorial for understanding A* is available from RedBlobGames (2016).

3.2.2 Heuristic Function

To estimate the cost to the goal we use what is called a heuristic function. For the algorithm to find the optimal solution the heuristic needs to have two mathematical properties: admissible and consistent. It can be shown that when the search space is on coordinates in Euclidean space, the heuristic will be consistent and admissible, if it always underestimates the cost of getting from the current position to the goal. One underestimating function is the product of the straight line distance from the current cell to the goal and the minimum value of the cost function. For example, if the cost profile had its minimal value for flat terrain the underestimate would be the equivalent of walking on flat terrain in a straight line from the start to the goal, assuming no obstacles. The point is that there is the most efficient path that could be taken, so any real path in an obstacle environment will perform worse. A slightly different heuristic uses a different way of underestimating the distance. Instead of using the straight line approximation it uses the so called Manhattan distance; this takes into account that due to the discretization of the map and the octile neighbourhood used in the algorithm any path can only be a sequence of a straight and diagonal lines. Thus a more realistic underestimate is not the straight line distance, but the shortest path that goes straight, and then diagonally at 45 degrees. It can be shown that the shortest such path has a distance as outlined in Equation 3.4.

$$h_{diagonal} = \min(|x_i - x_b|, |y_i - y_b|) \quad (3.4)$$

$$h_{straight} = |x_i - x_b| + |y_i - y_b| \quad (3.5)$$

$$h_{distance} = \sqrt{2}\Delta r \cdot h_{diagonal} + \Delta r \cdot (h_{straight} - h_{diagonal}) \quad (3.6)$$

The Manhattan distance can then be multiplied by the minimum of the cost function. In the case of the energy cost, this leads to the following heuristic, first used by Johnson (2008) and Essenburg (2008):

$$h_{metabolic} = R \cdot h_{distance} \quad (3.7)$$

Where R is the minimum energy rate on Figure 3-4. This happens at a slope of -5.85 degrees on Earth, where $R = 1.504m + 53.298$.

3.2.3 Shortcomings of A*

Although A* promises optimal costs, it is only optimal under consideration that paths can only consist of segments with the orientation being a multiple of 45 degrees. This discretization due to the grid and the way the neighbours are expanded can still lead to paths that are 8% suboptimal, as shown in greater detail by Ferguson. The zig-zagging effect produced when zooming in on the path also looks unnatural to the eye, and could raise questions from the EV astronaut navigating the path. This problem can often be solved through a post processing phase. One type of post processing consist in smoothing the path: the Ramer-Douglas-Peucker(RDP)(Douglas and Peucker, 1973) algorithm reduces the amount of points needed to describe the path while staying within a given set error bound. Although this makes the path look smoother, it also has the potential to increase the path cost. Another solution has a different approach: find the furthest point p in the path from a given point for which there are no obstacles, and the cost of this traverse is less than the equivalent cost in the path to that point. This solution works great in flat terrain, where the shortest unobstructed path between two points is a straight line, but requires more computation in the presence of terrain and more advanced cost functions. Previous versions of SEXTANT incorporated the RDP algorithm to smooth the path; this has not yet been ported over to the most recent version of SEXTANT.

3.3 Rough Terrain Adapted A*

3.3.1 Challenges with Higher Resolution Terrain

As described further in Chapter 5, steep and rugged terrain combined with high resolution datasets proved a challenge for SEXTANT. Previously, SEXTANT had been applied in smooth and shallow terrain at resolutions of 1m and as low as 240m. Even if the underlying terrain presented significant irregularities, the lower resolution of the map filters out the steeper slopes. Therefore, the number of slope-based obstacles represented a small fraction of the map, even when the maximum terrain slope was set as low as 15 degrees.

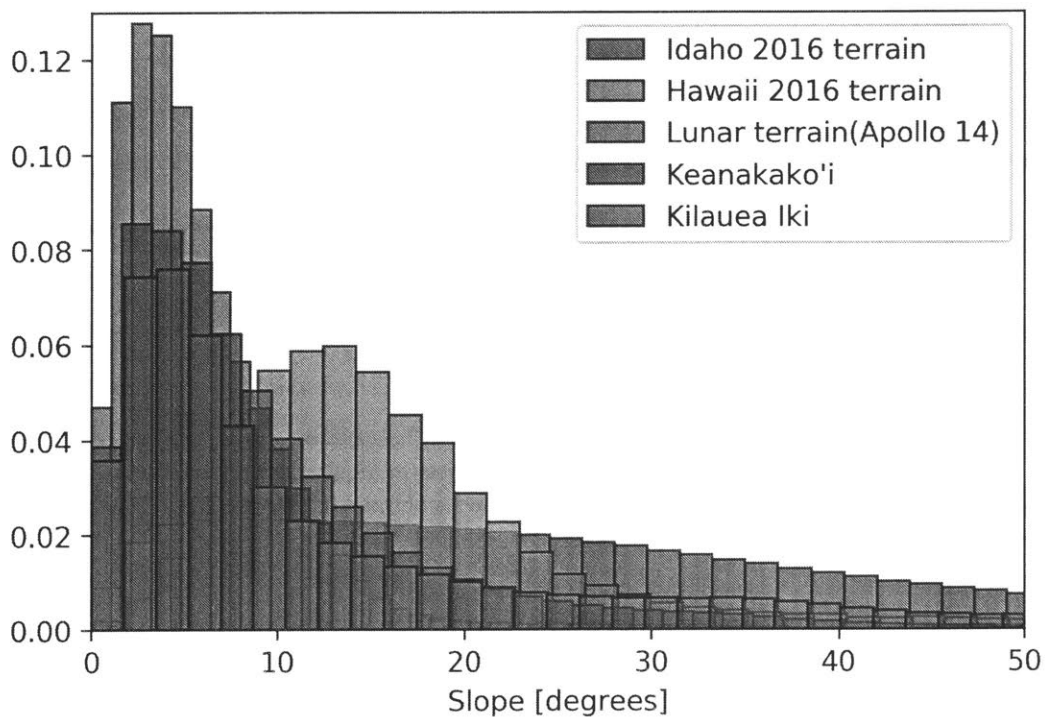


Figure 3-6: Slope Histogram

DEMs first encountered during the 2016 Idaho campaign offered a resolution of 2cm that captured the small scale irregularities in the terrain, this is shown in Figure 3-7. This introduced problems with the slope-based generation of obstacles, incorrectly classifying as obstacles smaller rocks and irregularities that could easily be

walked over during a human traverse. Combined with an already underlyingly steep terrain, the environment divides into isolated traversable regions of shallow gradient separated from one another by the prominent obstacle environment as displayed in Figure 3-7. The A* algorithm presented so far cannot explore beyond immediate neighboring cells; starting in an isolated region would quickly result in no path to the goal.

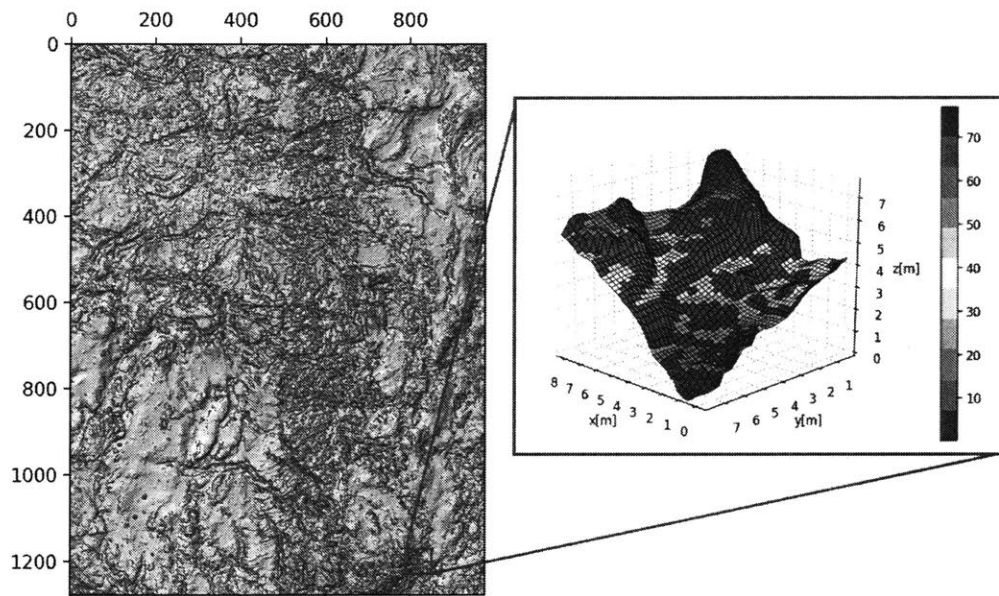


Figure 3-7: Challenges of rough terrain

Left: obstacle map from Idaho 2016 terrain draped on a hillshade representation of the terrain, with slope threshold of 35 degrees, red represents obstacles. Right: three dimensional representation of terrain, areas in red represent obstacles.

Three solutions to the large number of obstacles were considered: increasing the slope threshold, downsampling the map, and changing the path planning algorithm.

3.3.2 Increasing the Slope Threshold

The first solution is simple and requires no changes to the current method: increasing the maximum traversable slope will remove previously existing obstacles in the terrain expanding the search space for A*. However, this solution comes with several problems; although increasing the threshold could remove small locally slope-based

obstacles that the astronaut could easily traverse, it could also force the astronaut to take a path across consistently steep terrain, an unrealistic assumption. During application this method was limited to increasing beyond 35 degrees.

3.3.3 Downsampling the Terrain

Downsampling the terrain is equivalent to applying a low-pass gaussian filter to the elevation matrix representing the DEM. This is a well understood technique in many domains, and it removes irregularities at a small scale, reducing the number of incorrectly classified obstacles. Meanwhile, the technique also comes with an undesired effect: the smoothing is global. While irregularities are removed, the magnitude of the slopes is also reduced in larger steep areas. This could result in erroneous classification of the terrain as traversable. The obstacle washing out effect from downsampling is illustrated for a terrain on the Mauna Ulu Volcano cone in Figure 3-9.

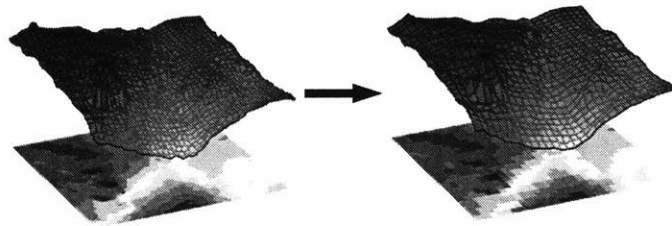
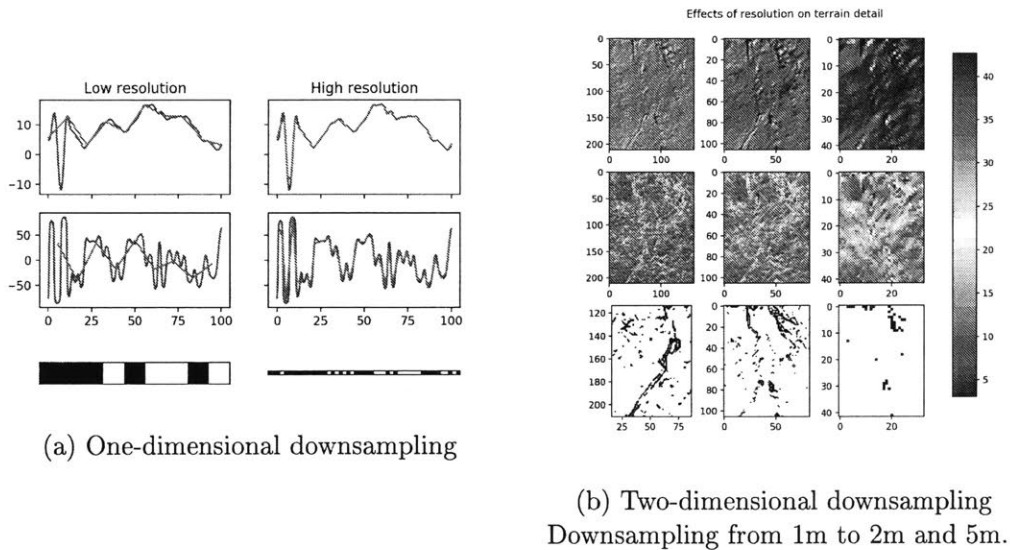


Figure 3-8: Solution 1: Lowering the resolution of the dataset

Figure 3-9 shows how the terrain is washed out, and the obstacles disappear, as the resolution is lowered from 1m to 5m. Since the path planning algorithm has no understanding of the underlying dataset describing the terrain, applying the path planner to the lower resolution maps might result in paths that traverse areas that in reality are obstacles.

The problems encountered by reducing the obstacle threshold or downsampling the terrain are similar: correctly classifying traversable irregularities as obstacles might lead to incorrectly classifying steeper terrain as traversable. More advanced custom techniques could be implemented to only remove obstacles at a local level; a



(a) One-dimensional downsampling

(b) Two-dimensional downsampling
Downsampling from 1m to 2m and 5m.

Figure 3-9: Downsampling effects

Sampling at different resolutions yield different obstacle maps, in this case for one dimensional slopes. In blue is the underlying simulated terrain, which was generated by building a random spline with *low frequency* variations(i.e. the terrain) and *high frequency* variations(i.e. rocks or other small variations). In orange is the sampled terrain points.

significant difference between small scale and large scale irregularities is the magnitude of the elevation offset from the locally averaged elevation. Only steep terrain with large magnitudes would actually classify as an obstacle. This technique should be further investigated. Next we present a different approach, which it the one currently implemented in SEXTANT.

3.3.4 Increasing the A* Search Space

The last method involves altering the pathfinding algorithm. Currently the algorithm only considers paths that move to neighboring cells of the current location so far in the path. Here we call the set of neighboring cells the search kernel(this definition is only used in this paper, as there doesn't seem to be any conventional name). This is a similar discretization to the one proposed by Galin et al. (2010). We propose search kernels based on pixelated approximations of circles, as illustrated in Figure 3-10b, the idea being that the algorithm can look beyond the immediate neighboring cells and skip obstacles, as in reality a person can choose different step length to overcome

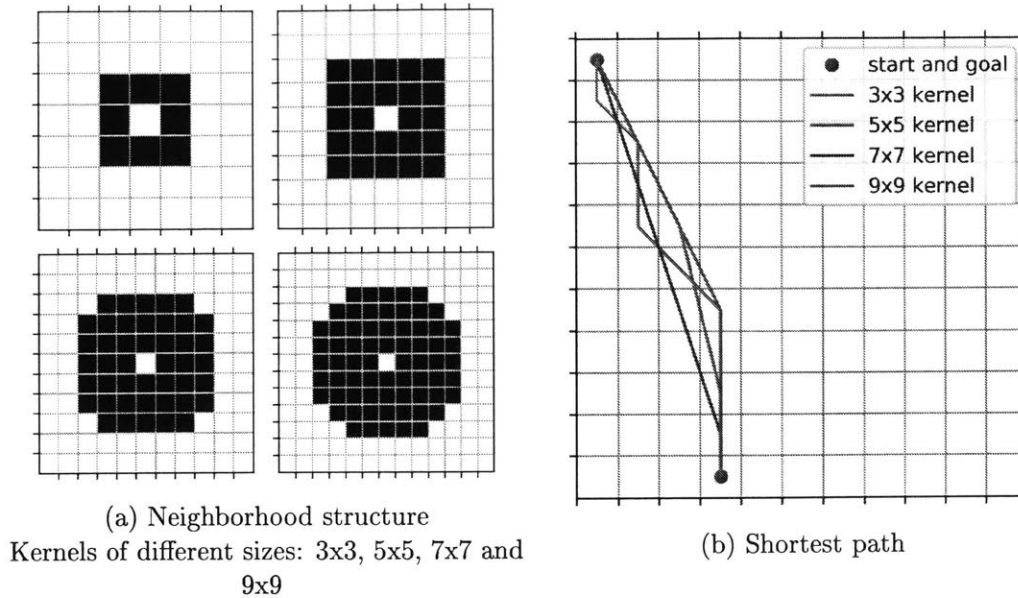


Figure 3-10: Search kernels

There are no even size kernels (such as a 4x4 kernel) due to the symmetry needed.

small obstacles. One way to think of the search kernel is that for every cell, the A* algorithm determines if a small immediate step can be taken (3x3 search kernel), or if a slightly longer step can be taken, and thereby potentially overcome an obstacle. This approach works when the DEM grid cell size is significantly shorter than the maximum step length for a human, on the order of a meter.

This method also comes with a second advantage: it considers paths at other discrete angles than multiples of 45 degrees similarly to other path planning algorithms as discussed in section 3.2.3 (on downsides of A*). As illustrated in Figure 3-10, this yields paths that might be more optimal, and that look more natural to the EVA astronaut.

However, the gain of overcoming obstacles and generating more optimal paths can come at the sacrifice of computational performance. Although in an obstacle free environment it can achieve faster solving times (since it can take large leaps towards the goal), in a general obstacle prone environment this will be the case less frequently, as the search algorithm has to expand a lot more options at every iteration. This is further discussed in next chapter.

Chapter 4

Performance and Results

4.1 Path Planning Performance in Rough Terrain

Datasets have gotten larger since the work of Marquez (2007). Larger DEMs, combined with larger search kernels, significantly increase the search space for the A* algorithm, and deteriorates the solving time. One of the observations from users interacting with SEXTANT, is that the waiting time can lead to uncertainty on whether the planner is just taking a long time to find a solution, or whether something stopped it from working incorrectly. A valid reason that could make the A* spend significant time resources before it eventually realizes that there is no solution is a case where the goal is surrounded by obstacles on all sides. This could be visualized as a “landlocked” goal. In this case, the A* algorithm will have to search the entire search space before it can decided that there is no solution. This problem could be resolved starting the search from the start and the goal; this is not a novel idea, and is used in other path finding algorithms such as D*(Koenig and Likhachev, 2002), and was mentioned by Essenburg (2008) as a previous incorporation into SEXTANT. Here we develop a different approach, if we instead can anticipate a general time performance as a function of a certain size map. This raises a question that previous work on SEXTANT had not addressed: time performance of the algorithm, which is conventionally denominated time complexity. Evidently the computation time will largely depend on the amount of obstacles, yet establishing a correlation with size and roughness of the terrain can

give users a general estimate for how long certain queries should take.

Unlike many algorithms, it is hard to bound the time complexity of the A* algorithm with a theoretical expression, as this depends very much on the exact heuristic function being used. The simplest solution is to determine the time performance empirically.

Time complexity also introduces the question of how to speed up A*. One simple solution is to shrink the search space by downsampling the DEM, as already described in Section 3.3.3. As was already described, this introduces potential problems as critical obstacles might be smoothed away. Performance can also be gained from using more advanced path planning algorithms such as introduced in Chapter 2 but not implemented in SEXTANT. Next we introduce another method from the literature, which trades time complexity with optimality of the cost function.

4.1.1 Speeding up A*

There is one simple way that we can make A* run even faster, by sacrificing optimality, as discussed in the Anytime A* paper by Likhachev et al. (2005). The idea is that *a solution* is better than no solution, even if it is sub-optimal. The idea is that by multiplying the heuristic function h from 3.4 by a weight, we use a heuristic that is no longer admissible, hence it can't guarantee optimality, however it has a tendency to give low cost estimates for neighboring cells that are in the general direction of the goal, while high estimates for neighboring cells in the opposite direction, hence accelerating the speed at which the algorithm moves towards the goal. This version of the algorithm will still make sure to avoid constraints, however it will just not explore most of the terrain and only explore cells which brings it closer to the goal.

4.1.2 Time Complexity versus Optimality

To compare the performance of the A* algorithm under the two methods we can use to speed it up, it was run on the highest resolution dataset we had, from the Idaho 2016 site. The comparison was done between the baseline 20cm resolution dataset(which

was downsampled from the original 2cm dataset), while running SEXTANT with a kernel size of 7(making the maximum step $3.5 \times 20 = 0.70\text{m}$) - which was required because of the large amount of obstacles, and different heuristic weighting factors. Then the dataset was reduced to 40cm resolution, and a kernel size of 5(making a human step size of around $2.5 \times 40 = 1.0\text{m}$), and to 1m and 2m, with a kernel size of 3 since this is the minimum kernel allowed. A comparison was drawn on both the time to run, and the optimal cost. The results from these are outlined in Figure 4-1.

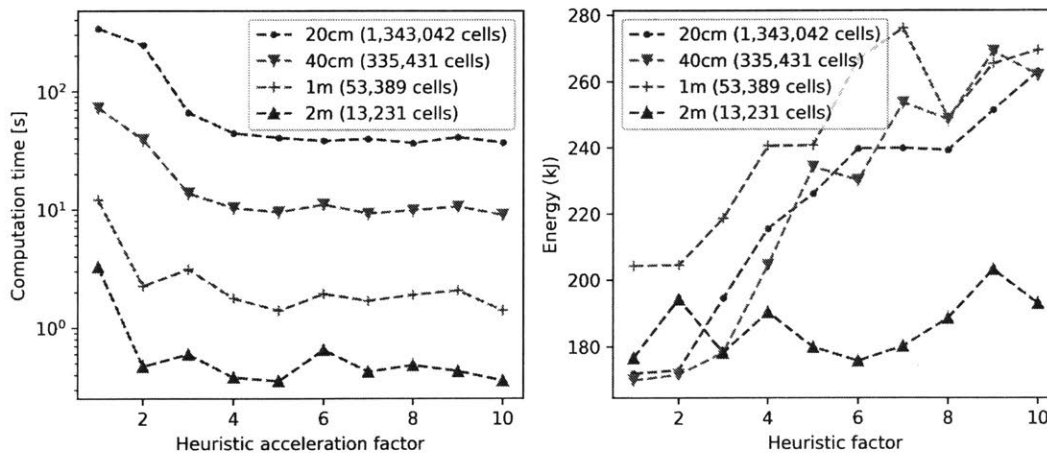


Figure 4-1: Time complexity versus energy cost as a function of different heuristic factors and map downsampling resolutions

As can be seen, the time complexity of SEXTANT seems to be linear with the search space, i.e. the grid size: going from above 0.2 seconds for the 2m DEM with 13,231 cells to above 200 seconds for the 20cm DEM with 1,343,042 cells. This would suggest that if the algorithm had been run with the 2cm DEM, it might have taken two more orders of magnitude(20,000) to solve with A*. As expected, the search time is reduced thanks to the heuristic factor, and can be reduced by almost an order of magnitude, but there seems to be a limit to how much can be gained. This makes sense: although the heuristic reduces the search space by ignoring options that might lead to a better path, it can only reduce the search space to a certain point; although the algorithm might not find the best path to the goal, it has to find at least a path to the goal.

In terms of minimizing the cost function, as expected, increasing the heuristic

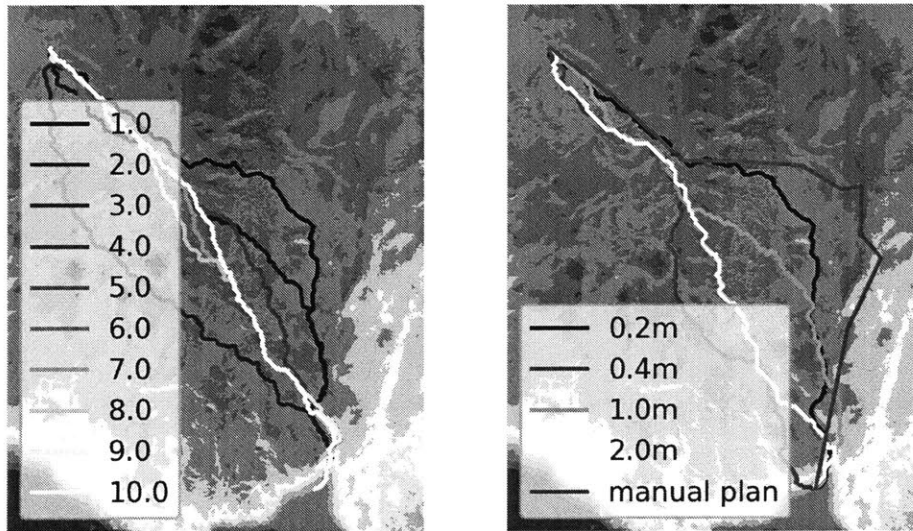


Figure 4-2: Comparing resolution and heuristic effects on path

factor in general has the effect to increase the cost. However, for the 2m resolution map there seems to be some unexpected behavior, but this is not further investigated. We also notice that the performance of the path on the lower resolution maps seem to outperform the cost of the higher resolution. Cost-wise it is easier to find a more optimal cost when running on a lower resolution map: many of the obstacles that might prevent a path to go through a certain area at a higher resolution might have been smoothed out.

We can plot the solutions on the map, to understand how the path differs. The solution paths are also significantly different, as can be seen in Figure 4-2: higher heuristic weights result in much straighter paths (since we are only aiming for the goal, and ignoring less costly paths), while lower resolution data seems to yield a similar result. This should be further investigated.

Next the thesis goes into a more applied section, where the application of SEX-TANT to BASALT are discussed.

4.2 BASALT Mars Analog Campaign Application

4.2.1 Overview

BASALT is a NASA research project exploring new operational concepts and hardware systems within the limited scope of Mars EVA for science and exploration. To this goal, three analog campaigns, consisting of ten mission days each, were carried out by analog astronauts and a large support team. The first, during the summer of 2016, was carried out in Craters of The Moon National Park in Idaho, the second and the third, during the fall of 2016 and 2017, both carried out in Hawai'i Volcano National Park, on the Big Island of Hawai'i. The work described in this paper was part of the 2017 campaign, conducted in the Kilauea Iki crater and Keanakako'i Overlook area.

BASALT's high-level architecture separates the mission teams into two EV astronauts and two intravehicular (IV) crew members that can communicate in real time on what is referred to as "Mars time". Additionally, there is a science support team (SST) operating on "Earth time" that can only communicate under delays. EVA missions extend over the period of 4 to 5 hours and are split into several phases with different durations: approach, contextual survey, sampling site selection, presampling survey, and sampling. The navigation is mostly of interest in the approach phase. The approach phase covers the traverse from the starting point of the EVA to the vicinity of a general area of scientific interest. During the 2016 campaign, the area of interest consisted of up to three science stations in close proximity of one another, during 2017 campaign it remained a significantly large area. For the 2016 deployment, the second and third stations were normally within a close distance of the first station, making the traverse planning and navigation mainly a challenge for the approach phase, which would generally last about 20 minutes. Compared to Apollo and previous analog missions, the traverse paths for BASALT were significantly shorter, with the longest traverse being around one kilometer, compared to 1.5 km traversed during Apollo 14. Although the traverses were short, they were executed in challenging terrain with slopes and more challenging terrain texture than encountered during

Apollo. The terrain encountered could vary from flat and solid to rugged and unstable. The representative terrain is depicted in Figure 4-3. This latter type of terrain included a'a, a type of cooled lava forming small sharp rocks, and shelly pahoehoe, a smoother looking terrain, but crusty, and with a large chance of collapse under the weight of a person. Therefore, the challenging terrain still made the approach an important part of the mission, where path planning and navigation were critical to the success.

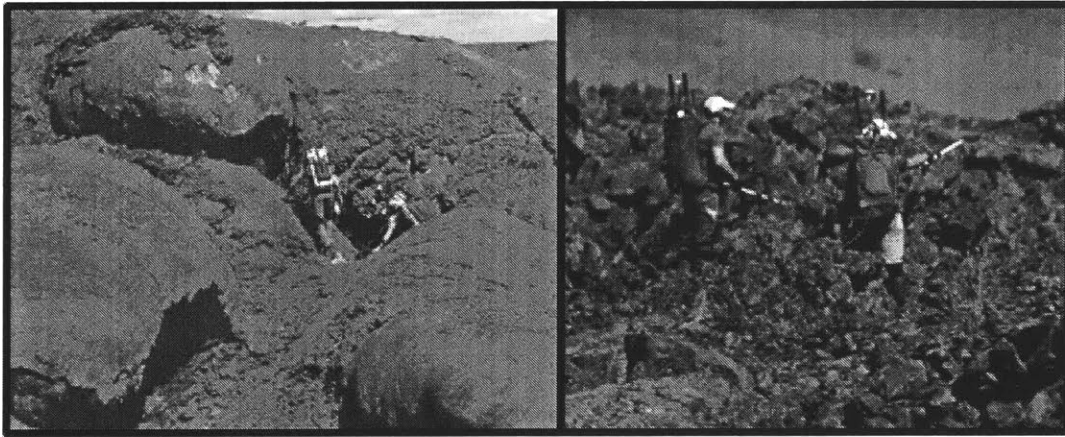


Figure 4-3: Terrain encountered during BASALT deployment

4.2.2 SEXTANT application use case during BASALT

Although BASALT EVA durations were on the order of several hours, only a smaller portion of the EVA consisted in exclusively traversing: the approach phase from the starting point of the EVA until the area of science operations. During BASALT, SEXTANT was used to plan the approach traverses, which could range from several 100 meters up to almost a kilometer. The portion of the EVA that included science operations and exploration consisted of regions small enough that the EV astronaut crew could easily find their way by eye without the need of a decision support such as SEXTANT. The approach paths were planned a-priori of the EVA, normally the day ahead, but replanning had to be executed once during the 2016 campaign mid-mission.

4.2.3 Path planning through xGDS

As outlined in Deans et al. (2017), SEXTANT was integrated in the exploration ground data systems(xGDS) developed at NASA Ames(Lee et al., 2013), which has been used for planning, executing and post processing the EVs. The advantage of embedding SEXTANT within their user interface is that this puts the planning capability at the fingers of the EV, IV and science backroom team during the several phases of the project. During the planning phase, it allows scientist to select the areas where they want to go during the deployment, and automatically plan the traverses which can give them further information on the time required for moving. During simulations, it allows IV to potentially replan the mission, this was executed once during BASALTs Hawaii 2016 mission.

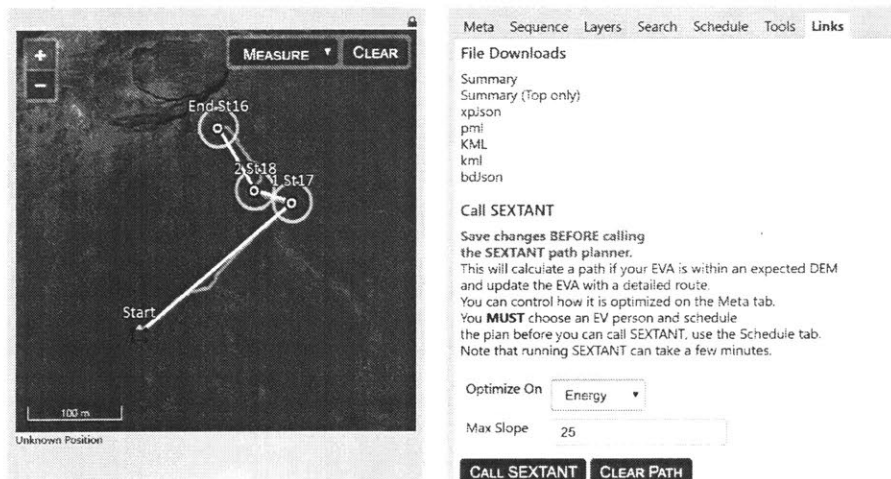


Figure 4-4: Planning with SEXTANT in xGDS

Users could add waypoints (displayed as white circles joined by yellow lines), which would normally correspond to science stations for the BASALT EVA. Next, the analog astronaut crew is selected, and their weight is used for the energy cost model developed. A simple interface allowed the user to choose the cost function to optimize - this was set by default to Energy, and Energy is the only cost function that was used during the BASALT campaign. Finally, a max slope could be set; it was set to default to 25, but at times changed to all the way to 35 to find a path if there were too many obstacles for the 25 degree max slope.

The interface, although very basic compared to SEXTANTs interface developed by Johnson et al. (2010a) allowed for a simple integration into the greater planning framework. On the downside, the map did not display an obstacle map as a function of the max slope, neither did it display any visualization of the cost map, such as discussed in the work by Marquez (2007). This made SEXTANT more of a fully automated tool than a decision support aid. Given the previous research on SEXTANT, this should be adapted in the next iteration.

4.3 Path Analysis

An obvious metric to measure the quality of the plans generated with SEXTANT is the root mean squared offset between the planned and executed paths. Significant deviations from the path could be interpreted as the EV crew redirecting the path due to challenging terrain or obstacles. However there could be other causes for deviation; interesting features in the terrain could lead the crew to explore in a different direction than that dictated by their path. It must also be emphasized that the EV crew uses the path as a guideline, and do not follow it and that the offsets are sometimes based on improvised local navigation, given the fact that the map used to plan had a limited resolution.

Although SEXTANT's main application has been finding optimal paths, its predictive ability is also of interest, with the capability to estimate traversal times, distances, and energy cost. These metrics can then stand as a good benchmark for the performance models used in SEXTANT on which they highly depend. Therefore, SEXTANT predictions are compared to actual metrics with the exception of the metabolic cost, for which no data was gathered. The traversing phase includes times when the crew is stopping, or wandering in the vicinity of the path for a short while if any location of interest is spotted. Since the metric of interest is only the pure traversing time, GPS data from the EV crew was manually filtered to only include the times the crew is traversing (walking at a steady pace towards a goal, and not wandering constantly in the same area).

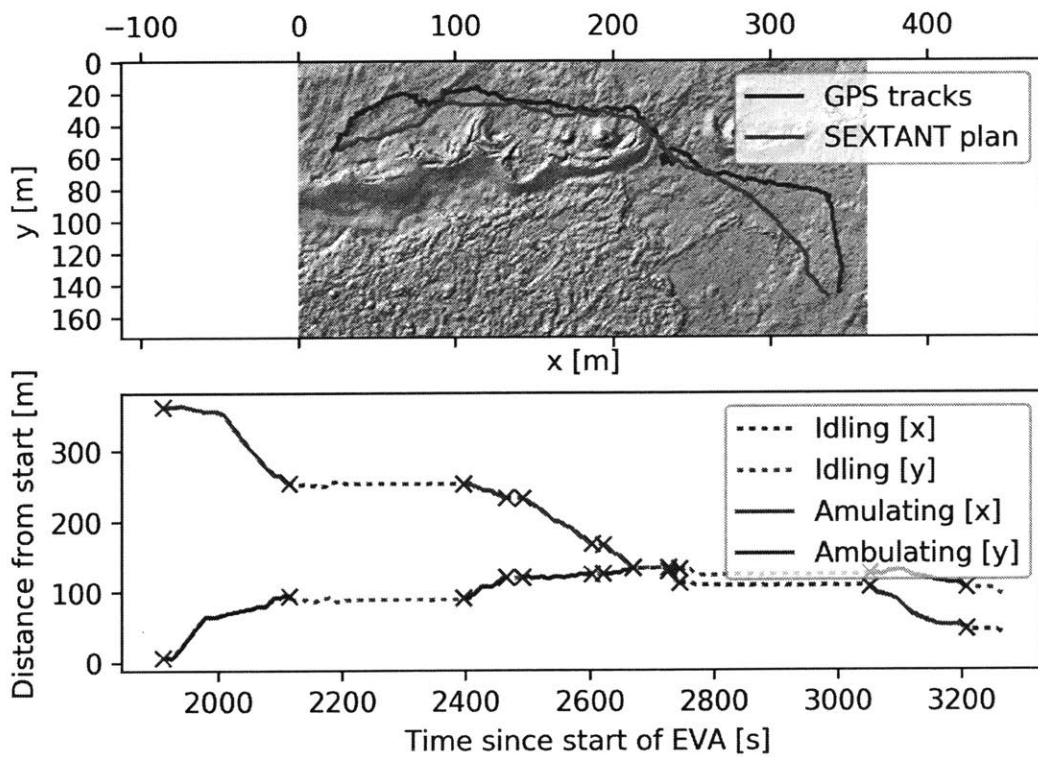


Figure 4-5: Example of GPS processing of EV tracks
 Dotted lines represent sections in the time line that were not considered when calculating the total time and distance metrics.

To analyze the GPS track, the signal was first filtered to remove sections when a person is mostly stopped, since this would add extra time to the approach travel time estimate. This was done manually for all tracks. Then the partial times were added up to compute the total approach time, and the distance was only added up during the time of traversal. Although noise in the GPS signal could add distance, the signal to noise ratio was assumed to be significant for the ambulating sections, which can easily be seen in the data.

4.4 DEM sources

Several DEMs were used for the path planning effort during the BASALT campaign. These are all listed in Table 4.1. During the Idaho 2016 deployment at the Highway lavaflow in Craters Of The Moon National Park, no topographical data was available until the last day of the mission, when a 2.11 cm drone scan was completed. This is the most detailed map that was gathered during the entire mission, leading to a file of size 28,247x35,534 pixels. Although the planning region during Mission Day 10 of Idaho was 12,441x9,624, this is still a very significant size. To reduce data, this DEM was downsampled to 20cm, which was assumed would not result in any significant loss of details (although it was not reduced further, as discussed in Section 3.3.3).

Table 4.1: Dataset sources for SEXTANT during BASALT

Case	Map Size	Resolution	Source
Idaho 2016	28,247x35,534	2.111cm	Drone scans
Hawaii 2016 full	8,313x9,332	0.48 m	Stereophotography from satellite im- agery
Hawaii 2016 post	2,079x4,157	1m	Airborne LIDAR
Hawaii 2017	7,493x4,016	1m	Airborne LIDAR
Hawaii 2017	2,993x1,262	0.5m	Ground LIDAR

The Hawaii 2016 deployment around the Mauna Ulu volcano in Hawaii Volcanoes National Park used a DEM that was generated through ASP as described in Section 3.1.2. Two satellite images were used as stereopairs, and kindly processed

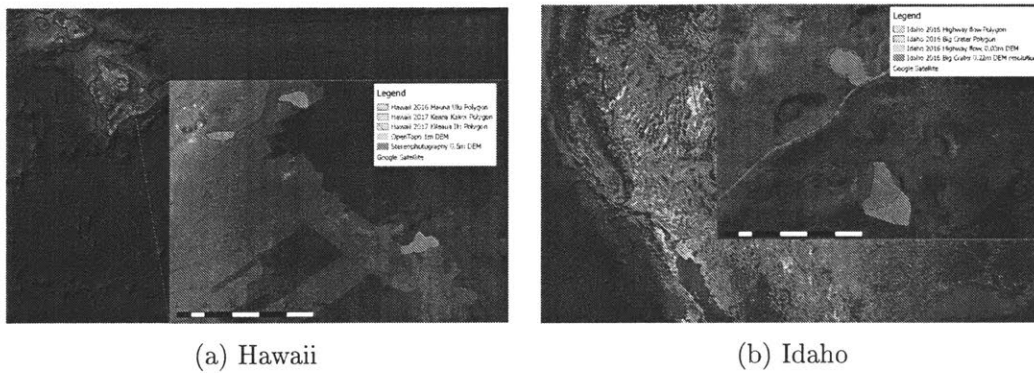
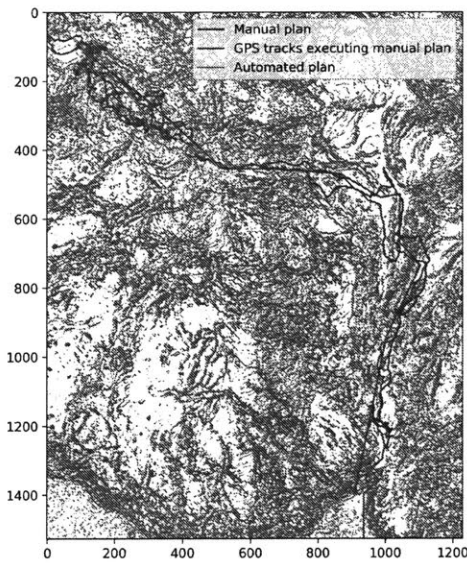


Figure 4-6: Overview of path planning map products used during BASALT campaign

by Christopher Haberle at Arizona State University, generating a 0.48m resolution map from the 0.3m resolution imagery. Although the DEM generated through ASP is representative of larger features in the terrain, it can sometimes have a hard time capturing smaller details. This was discovered when after the 2016 deployment, a second DEM with much higher accuracy was discovered. An aircraft LIDAR survey of the Kilauea crater area of Hawaii had generated a 1m resolution DEM with much better accuracy (NCALM, 2007). In this section we first present the results from planning with this DEM, and later we compare the results of planning in the same area with two different DEMs - this shows the differences that can arise when using maps that do not accurately capture the map at the human scale.

Finally, for the 2017 Hawaii deployment back to Volcanoes National Park, two sites were used, the very flat region of the Kilauea Iki crater and the more rough terrain in the Keanakakoi. The terrain encountered in Kilauea Iki didn't really require any automated path planning efforts, as the flat terrain made it simple for a person to find their way. Both terrain maps were also extracted from the same airborne LIDAR dataset that was used after the fact for the Mauna Ulu region. Finally, one of the BASALT members mapped certain areas during the 2017 deployment down to a resolution of 0.5m first, and then 0.1m. However, the data was very noisy, and post-processing was dropped since a good enough map existed already. Figure 4-6 shows the datasets used during all BASALT deployments:



(a) Comparing manual and automated
In gray are areas with slope above 25 degrees
which was set as the threshold maximum
slope



(b) Slope map used for manual planning
Green indicates slopes below 25 degrees,
yellow between 25 and 35 and red above 35
slope

Figure 4-7: Idaho 2016 plan for Mission Day 10 with high resolution terrain

4.5 Results

4.5.1 Idaho 2016

During the Idaho 2016 deployment at Craters of the Moon National Park, path planning was executed manually due to the lack of DEMs with enough resolution to capture the roughness of the terrain. When the drone generated terrain map was available, the rough terrain led to the problems discussed in Chapter 3, preventing automated planning from being used. The high resolution combined with the rough terrain made it challenging to generate the plan manually, as there are a lot of obstacles, and the problem becomes similar to that of solving a maze. In the end it required almost 2 hours of manual labor, versus a few minutes when planned automatically. The automated plan was generated after-the-fact, by using a 7-type kernel, setting the maximum slope to 25 and the weight of the EV astronaut crew to 80kg, and only giving the first and last point as a waypoint.

This example also show how the automated planner managed to find a slightly

shorter path than a person; yet there seems to be some agreement, especially towards the end of the path, as the automated plan agrees quite well with the manual plan. To compare performance of the manual and automated plan, SEXTANT was used to plan the intermediate path between the waypoints the manual planner had given. The same settings were used as for the plan from start to target, and the energy cost of both plans are compared:

4.5.2 Hawaii 2016 deployment

During the Hawaii 2016 deployment, SEXTANT was used for mission days 2 through 9, mission day 1 still relied on manual planning, and mission day 10 was canceled. As discussed in more detail in Section 4.4, the map used had been generated with the Ames Stereo Pipeline. All plans were generated with a 25 degree maximum slope threshold, and with a kernel size of 3, as the terrain map did not offer the same type of ruggedness as observed during Idaho, and also due to the resolution being almost on the scale of a human step. However, the terrain did have significant rugged features that hinted towards a poor representation by the map. The terrain also presented several artifacts that are typical from the ASP generation process; gridded stripes can be seen when looking closely at the hillshade visualization of the terrain. This indicates that before using SEXTANT, the terrain map should be verified, at least to a qualitative extent - during the Idaho deployment this was not necessary, as the terrain topology had been very accurately gathered through means of a drone.

Figure 4-8 shows all plans in solid lines, and GPS tracks from the EV crew that was navigating the path. The last three stations normally constitute the science stations where most of the EVA is concentrated, explaining the high density of tracks in this region. If we however focus only on the track between the first two or three points, which normally constitutes the approach phase of the EVA, we can see that some tracks have more significant discrepancies than others. At times this could be from the EV crew wandering off, although for Mission Day 4, there seems to be a significant detour that had to be made. The EV crew would be highly satisfied with the paths generated by SEXTANT on some days, an highly unsatisfied on other. The

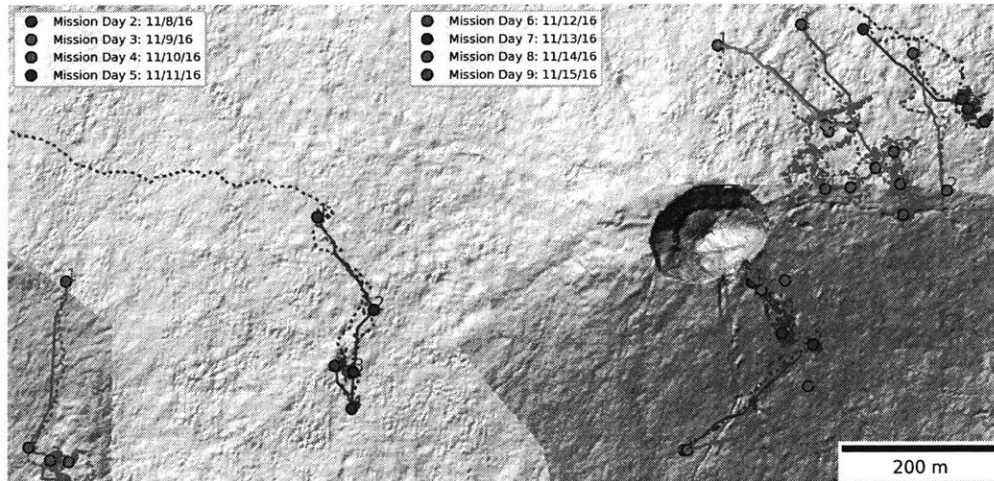


Figure 4-8: Overview over Hawaii 2016 deployment, mission days that used SEXTANT

most likely source of this is the accuracy of the map.

4.5.3 Hawaii 2017 deployment

The Hawaii 2017 deployment was split into 3 days in the Kilauea Iki area and 7 mission days in the Keanakako'i area. For this deployment the 1m resolution LIDAR generated DEM was used, giving rise to very accurate topography maps, as can be seen through the hillshade visualizations in Figure 4-9 and Figure 4-10. Similar to during the Hawaii 2016 deployment, plans used a kernel of size 3, and a maximum slope of 25 degrees.

The Kilauea Iki area was very flat, and even had a trail that went through the middle of it. This meant that most paths planned would be straight, but also that it would have taken a person no time to find the same solution, even in the field. The resulting paths are displayed in Figure 4-9. The deviation from SEXTANT's path during Mission Day 7 comes from the trail that already exists in the area, which the GPS tracks follow.

The Keanakako'i area displayed a greater range of terrain ruggedness, but most of the paths could be followed closely; paths from Mission Days 8 and 9 used an

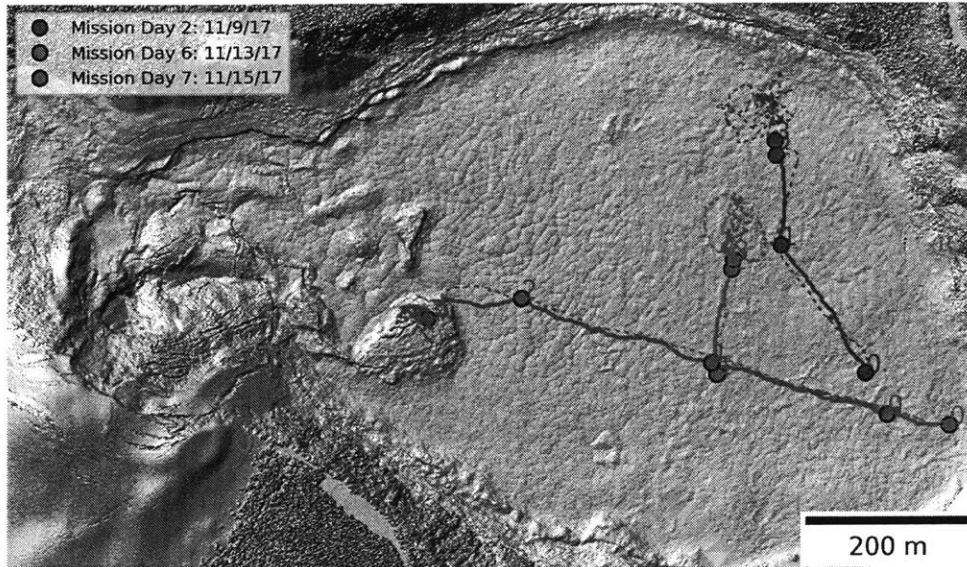


Figure 4-9: Overview over Hawaii 2017 Kilauea Iki site, and plans produced by SEXTANT

augmented reality display further discussed in Chapter 5 for navigation, which made it much easier to navigate the path and follow it as closely as possible during the approach phase. Planning for mission day 8 gave some insight into the advantage of SEXTANT: planning crew were convinced they could enter the region they were aiming at by going straight, but SEXTANT showed that there was no such path, and that they would have to contour the area and follow a ridgeline before entering the region from a different side.

The GPS tracks were processed as described in Section 4.3, generating Table 4.2 and Table 4.3. The reason the distances were measured were to justify discrepancies in the time to execute the paths; as the EV crew walk faster, or find a shorter path, or spend time exploring their environment, troubleshooting a problem forcing them to stop, or any other distraction from walking the path, it should affect the path length significantly, and thus the ability to predict the time. One of the key observations from comparing the estimated EV path times and the executed times, is that currently the planner always underestimates the time. This almost certainly

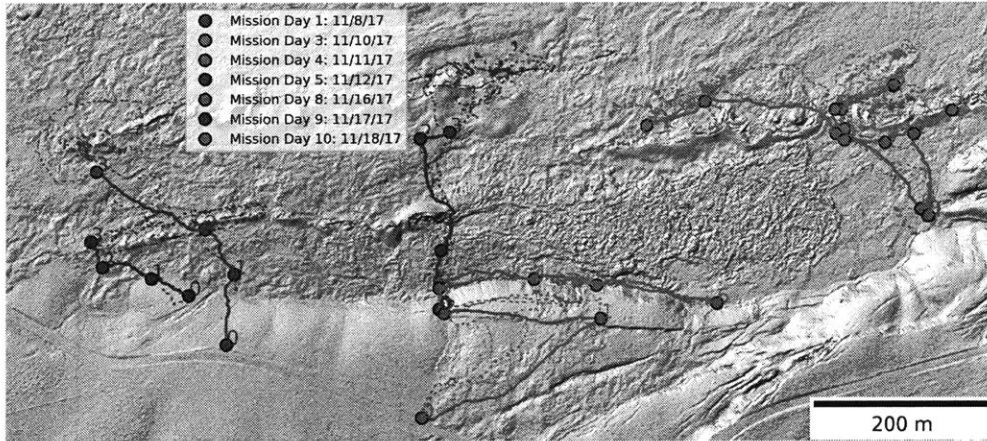


Figure 4-10: Overview over Hawaii 2017 Keanakako'i site, and plans produced by SEXTANT

comes from an incorrect velocity model, that overestimates the velocity at which the EV crew moves. An extreme example is Mission Day 7, where the executed time is 4.26 times the estimated time. This data suggest the real need to update the velocity model used by SEXTANT. This is further discussed at the end of this chapter.

Table 4.2: Comparing planned and executed performance

	1	2	3	4	5	6	7	8	9	10
Planned distance[m]	303.9	332.2	244	400.9	298.1	457.4	664.9	420.5	147.6	838.8
Executed distance[m]	231.8	342	285.3	455.4	405.9	509.1	780.7	431.1	145	1079.2
Factor	0.76	1.03	1.17	1.14	1.36	1.11	1.17	1.03	0.98	1.29

4.6 Lessons learned

Another key piece of information needed for traverse planning, that the BASALT deployments underlined, is information on the surface trafficability. The automated path capability was only based on altitude, and during the deployment, on several iterations, the crew had to walk in a different direction than the planned path due to

Table 4.3: Comparing planned and executed times

	1	2	3	4	5	6	7	8	9	10
Planned time[s]	191	208	153	252	187	287	417	264	93	526
Executed time[s]	350	418	451	610	796	433	816	626	151	2221
Factor	1.83	2.01	2.95	2.42	4.26	1.51	1.96	2.37	1.62	4.22

the presence of material that could be considered a safety hazard, even though the terrain was flat, and a from looking at the topological map the area looked safe to traverse. In the example of BASALT, two types of terrain were particularly difficult: shelly pahoehoe and a'a. Shelly pahoehoe consists of lava crust that can break under the weight of a person, creating the potential of falling. This is the reason of the alternative route taken by the EV crew seen in the background in Figure 4-11. A'a, on the other hand consists of pieces of lava the size of rocks, with very sharp and jagged contours, which will easily cut through clothing and other soft material. A'a is also not very stable to walk on, as the pieces of A'a can easily roll around or redistribute under the weight of a person.

Terrain type also affects the objective functions used for SEXTANT: two significant functions are the velocity with respect to the terrain slope, and the energy consumption per unit slope in the terrain. Currently the energy model built into SEXTANT relies on velocity functions taken from the lunar traverses and developed by Marquez, and the energy consumption models are based on walking on smooth treadmill surface at different angles. An attempt to improve the velocity curves from establishing a correlation between the slope and the ambulation velocity of the extravehicular crew across all Hawaii traverses can be found in Figure 4-12. Each dot corresponds to the instantaneous velocity at a certain location on the map, where the slope was calculated interpolating EV crew GPS positions from the DEM. Such analysis is bound to be very noisy, as there is a certain uncertainty in the GPS, uncertainty in any bias offset between the GPS and the DEM, and there was no recorded data on the properties of terrain traversed. Most importantly, however, is the correlation with factors that are completely independent from the slope, such as crew walking

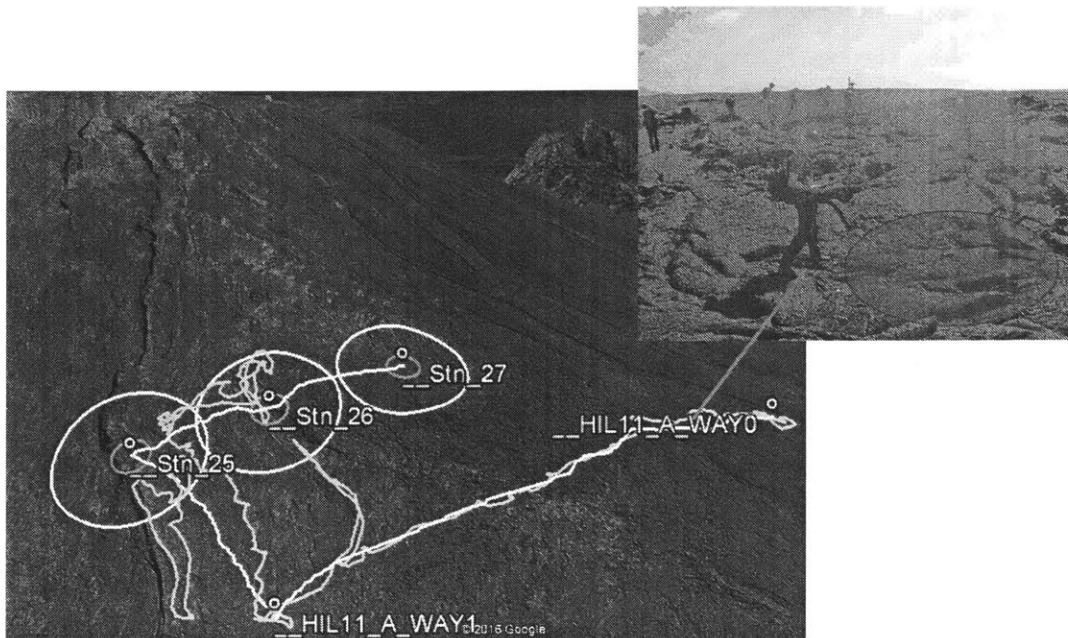


Figure 4-11: The importance of terrain

SEXTANT path taking user through shelly Pahoehoe, which can easily crumble underneath the weight of a person. In the background, path taken by EV crew to contour the Shelly Pahoehoe area

slower to better observe their surroundings.

Erroneous velocity or metabolic rate models result in incorrect energy and time estimates which are critical for the proper modeling of the traverse time. Changing such parameters might also skew the optimal paths towards different areas but could still be optimal even with incorrect underlying models, as long as the model captures the right trends, higher speeds at lower slopes, and the converse. However, any performance predictions based on such models would be incorrect - this was indeed documented by Gilkey et al. (2011) and was reaffirmed during the BASALT deployments, where typical traverse would take 20 minutes, where the times estimated by SEXTANT were on the order of 4-5 minutes as seen in the results.

A significant contribution in making SEXTANT more accurate would therefore come from having a larger set of velocity and energy curves for human walking in different terrains, under different loads. For analog missions such data would be critical from Earth environments, but for real missions to Moon and Mars, such data

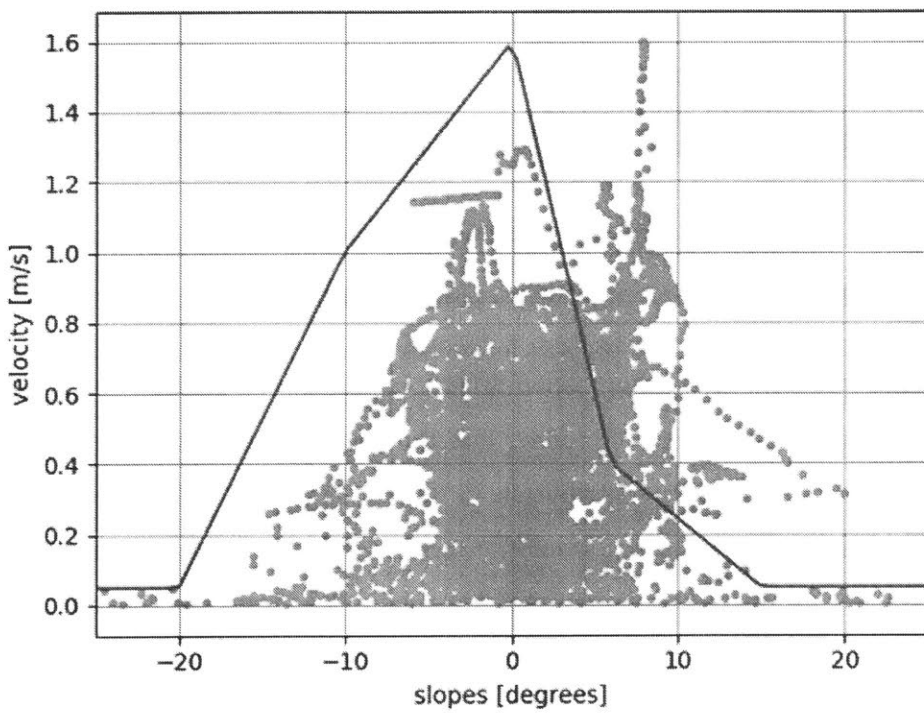


Figure 4-12: Velocity distribution
Velocity distribution from Hawaii 2016 GPS tracks

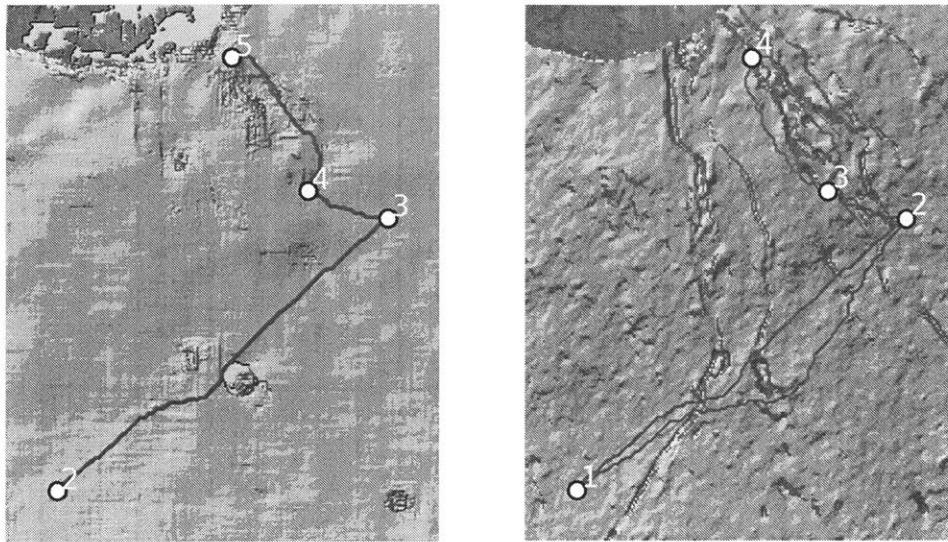


Figure 4-13: Planning with two different maps

On the left is the DEM generated with ASP from two satellite images. On the right is the LIDAR DEM. The plan using the first DEM is in blue, and in green for the second one.

should be gathered with off-loading capabilities to simulate the gravity conditions. This is already done with ARGOS today, but the testing should be extended to different terrains. Another key contribution in improving SEXTANT's predictions, especially for timeline management, would be to include better models that include margin to account for observation activities that might slow down the traversing.

The last important lesson is fairly straightforward but it is important to highlight it: how good a path is found by the path planner heavily depends on the underlying map. Figure 4-13 shows two paths with the same waypoints and maximum slope settings, generated with two different maps: an ASP generated map (less accurate), and a more accurate LIDAR DEM. The plan is from one of the mission days during the BASALT 2016 deployment.

Chapter 5

Path Planning Interfaces and EVA Navigation Aids

5.1 Introduction

The path planning functionality is interesting as a standalone capability, yet without a planning User Interface(UI) it would provide limited utility for EVA traverse planners, and without a navigation UI it would not be useful for the EV crew following the plan. This section therefore focuses on the stakeholders involved with path planning, from the planners who provide the input to the path planning capability and can tweak the parameters discussed in earlier sections, to the EV crew who will take the output, i.e. the path, follow it in the field, and who might themselves also take part in a re-planning effort mid-EVA.

As a matter of fact, previous work leading up to SEXTANT, including Marquez's work on PATH, Essenburg's work on Pathmaster and Johnson's work on SEXTANT, have all made the implicit assumption that the functionality of a planner and the UI used to work with that planner, cannot be separated. As for navigation, Johnson also discussed integration through NASA Ames iMAS system(Johnson et al., 2010*b*). But none of the previous work formalized the separation between the path planning capability and the UI itself.

This section formalizes the separation between path planning capabilities, plan-

ning UI and navigation UI. This separation is reflected in the structure of this chapter: the first section explains the framework to integrate the path planning capability of SEXTANT into external user interfaces to provide a plug-and-play capability. The second section discusses two examples of path planning UI integration which were accomplished during the BASALT project: first with an integrated traverse planning environment, and second, with a combined path planning and navigation interface which was used during simulations in the field by the EV crew. The last section presents results from implementation of an augmented reality display navigation UI which was tested during the last deployment of BASALT.

5.2 SEXTANT Interfacing Architecture

SEXTANT, in its most recent version is built on top of Python, a non-proprietary programming language that has had much success in the scientific and engineering community, allowing people to pick it up quite easily. It supports both *Functional* and *Object Oriented* paradigms, which is essential to giving SEXTANT its flexibility. The python code behind SEXTANT has been denominated *pextant*, following Python's non-capitalized naming convention. Furthermore, the code is openly available through an online hosting service by the name of *GitHub*, and can thus be easily accessed and used by anyone. This lowers the barrier to entry, and extends the application to a much larger user base.

The *pextant* library includes several basic functionalities related to handling map data sets, converting between DEMs and TINs, visualizing the data, and easily post-processing the output. This library can then be used through Python to execute the planning taking advantage of a highly modularized and simple coding interface. The key functionalities of the API are described in the Appendix A.

This solution does require familiarity with the Python programming language. Therefore, a simpler interface has been developed: an Automated Programming Interface (API). The *pextant* API can be thought of as a black box which takes a set of input datasets and parameters, and produces a path plan as the output. Both the

API and the standalone library can therefore be used in an automated way to for example process a large set of inputs, without need of user input.

5.2.1 Pextant API

Pextant’s API consists of a black-box like function, taking several datasets and parameters as input, and producing a path plan as an output. This is displayed in Figure 5-1. The API is available either as a command which can be used through the command line on a computer with Python and the right environment installed, or as a network server call, which makes it possible to submit *pextant* queries over the network. This makes it especially interesting when SEXTANT might be needed as a remote service, such as a web application, which would only need to submit a call to a certain web address to receive a path. The capability could even be hosted at a larger scale as a web-service to make SEXTANT available to anyone, although this has not been done.

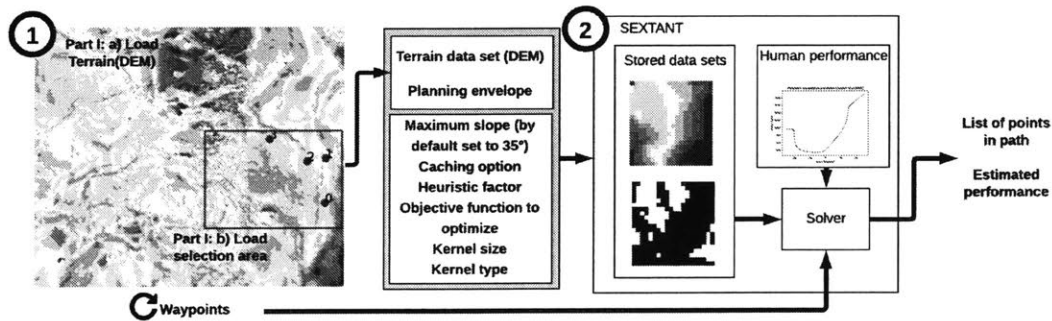


Figure 5-1: High level overview of the SEXTANT API

Inputs

The first set of inputs consists of: a DEM, generally in the form of a GeoTiff file, a planning area of interest envelope, which can be a subsection, or the full part of the map, the maximum slope allowed, the size of the search kernel, the goal to optimize(distance, time or energy), and two options to speed up the algorithm: the

caching option, and a speedup factor, which is related to the Heuristic function discussed in Section 4.1.1. This loads SEXTANT up, and it can then be further used for queries until shutdown.

One of the goal capabilities for SEXTANT was re-planning. As shown in Figure 5-1, to enable this, the inputs that go into SEXTANT have been split into two: the input that loads in the beginning, and the queries that each time a new path traverse is required can generate.

The purpose of having an envelope is to save on memory by only storing data on the specific area where the planning will be carried out. This data is then further post processed to calculate the obstacle map, and if caching is enabled it will further store all the calculations required for the modified A* algorithm.

The queries consist of a set of waypoints, for which the planner should be run. Loading and caching information ahead of time, and only using queries as input when SEXTANT is running, saves the time which previously was involved in loading all the data into SEXTANT for every query.

Outputs

Sextant outputs a path as a sequence of coordinates for the person to traverse. It also outputs information calculated during the run such as total cost function.

5.3 Planning Interfaces

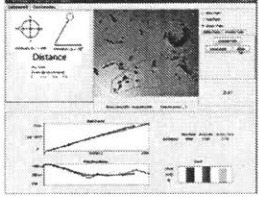
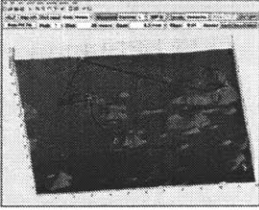

5.3.1 Past Interfaces

Table 5.1 shows the user interfaces implemented previously for PATH, Pathmaster, and SEXTANT, with details on the framework on which the UI was built.

As outlined in Chapter 2, Marquez's work focused on evaluating different map interfaces and whether there was any significant difference in manual planning performance, and as such, the user interface was designed with the user testing case in mind. UI work that followed for Pathmaster, and eventually SEXTANT did not

follow a rigorous UI design procedure, as the research focused on automated path planning, and not at building an optimal interface for manual path planning. Their main addition came in the form of a three dimensional map view, when ported to MATLAB by Essenburg (2008), which might be more intuitive from a user perspective to gain understanding on the terrain geometry.

Table 5.1: Previous traverse planning tools coupling computation and UI

Tool name	Obstacle criterion
PATH: Java based application (Marquez, 2007)	
Pathmaster: MATLAB based application (Essenburg, 2008)	
Original SEXTANT: MATLAB based application Johnson et al. (2010b)	

Next we discuss several other interfaces, two of which were used in the context of the BASALT campaign.

5.3.2 xGDS

As outlined in Deans et al. (2017), SEXTANT was integrated in the exploration ground data systems(xGDS) developed at NASA Ames(Lee et al., 2013), which has been used for planning, executing and post processing the EVs. The advantage of embedding SEXTANT within their user interface is that this puts the planning capability at the fingers of the EV, IV and science backroom team during the several phases of the project. During the planning phase, it allows scientist to select the areas

where they want to go during the deployment, and automatically plan the traverses which can give them further information on the time required for moving. During simulations, it allows IV to potentially replan the mission, this was executed once during BASALTs Hawaii 2016 mission. The use of SEXTANT combined with xGDS was discussed in Section 4.2.3.

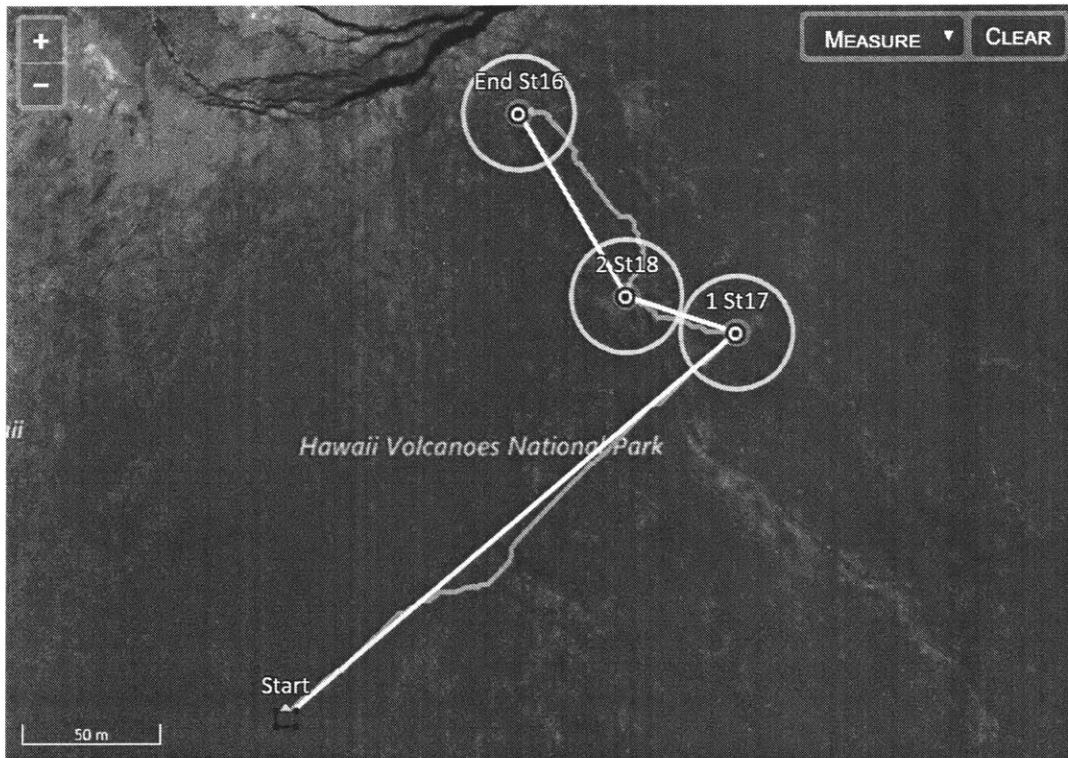


Figure 5-2: Planning with SEXTANT in xGDS

5.3.3 SEXTANT Web Application

A custom interface to use SEXTANT outside of xGDS was also developed. It was built on a web platform to make it easy to use, while removing any installation requirements on the user end. It was mainly designed to be used outdoors, similar to a previous setup by Johnson et al. (2010b). It allowed to test the re-planning feature for the first time, which revealed significant progresses that need to be made to speed the algorithm up. The SEXTANT webapp is discussed in greater depth in section 5.4.2.

5.3.4 Geographical Information Systems(GIS) interface

GIS tools are an obvious choice for viewing, processing and analyzing terrain data, which is one of SEXTANTs main point of entry. GIS tools also support drawing shapes and points, which would enable a simple way to draw waypoints. In fact, this was one of SEXTANTs original interface methods (Lindqvist, 2008), however the tool at the time was built specifically to interface with GIS but not more generally other tools. Since some largely used GIS tools(for example QGIS) come with python installed, it should be fairly simple to integrate SEXTANT into them.

5.4 Navigation Aids

The capability of automated traverse planning is only useful if the output can be followed by the extravehicular crew during their traverses. Navigating by landmarks, and guidance from the IV crew is of course an option, but offers its set of challenges, ranging from potential of miss-interpretation of guidance instructions to an added feeling of dependency.

In the context of planetary surface missions, we only have navigation experience from the Apollo missions. This pre-dates the availability of robust digital displays, and required the use of paper-based maps, as shown in Figure 5-3a. This required landmark-based navigation, and reference frame rotation by the astronaut to align their environment with the map, thus increasing the cognitive load. Navigation by paper-based maps and landmarks also proved to be challenging in certain occasions, as the second EVA of Apollo 14 proved: Commander Alan Shepard and Lunar Module Pilot Ed Mitchell, were headed on the last Lunar EVA without the Lunar Rover in the site of Fra Mauro. This involved a significant traverse of 1.5km to the rim of Cone Crater. However, on the way, they struggled to estimate positions to nearby landmarks such as craters and rocky features, which made it hard for them to triangulate their own position. After extending the EVA for 30min, they failed in finding the goal, missing their final target by less than a 100m. Due to the lack of atmosphere on the Moon, it is much harder for the human eye to judge distances, and thus triangulate

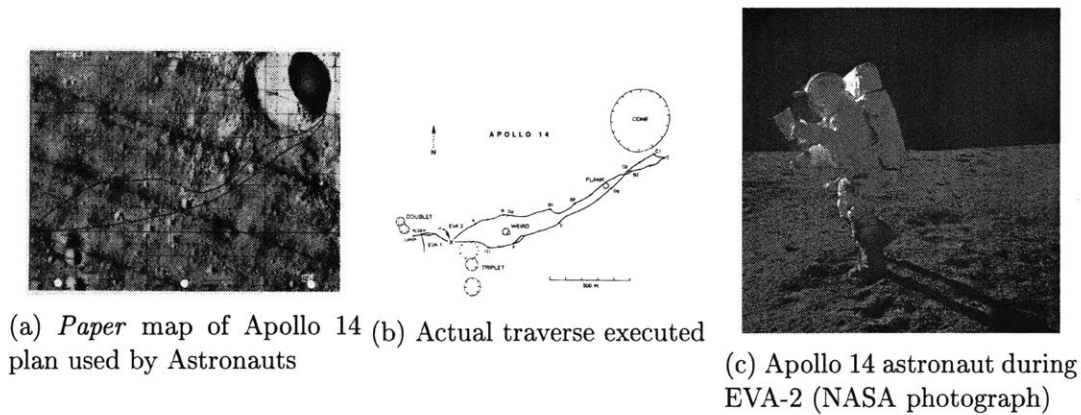


Figure 5-3: Navigation during the Apollo days

the position with respect to nearby landmarks.

This thesis discusses two navigation solutions, a direct screen display, such as wrist displays or hand-held tablets, which have been conventionally used in other analog simulations, and a novel display, based on Augmented Reality (AR), which has not yet been used for these purposes in the literature. This section however, limits itself to the description of the interface, the functionalities offered, how to use it, and the motivations behind the design. It does not go into rigorous user interface testing, however, we will bring up user feedback from the BASALT 2017 deployment, where the wrist display was used on almost all mission days, and the Augmented Reality display was used during two of the mission days, and in the same terrain as the mission, but out-of-simulation.

The main focus of this section is on tackling the navigation problem, but we will also discuss other functionalities from the same environment, and how these fit together.

5.4.1 Background

To inform navigation interface designs for Mars EVA missions, we must explore existing navigation tools, including paper maps and 2D digital interfaces. Human navigation, orientation, and positioning rely on a feedback loop that takes in various proprioceptive and environmental cues to build a mental model of a space and orient

themselves within this mental model. Humans rely on visual feedback, vestibular feedback, and kinesthetic feedback to position themselves within their mental model of a space(Bakker et al., 1999). Environmental features can serve to support or reject where a user thinks they are within their mental model, thus refining the accuracy of the mental model and organizing the relationship between said features. The process of building up a mental model of a space can be described as building a configurational schema(Couclelis et al., 1987; Golledge, 1999). Although using a configurational schema is a natural approach to navigation, it is not necessarily stable or coherent due to limitations in human memory. Furthermore, building configurational schema relies heavily on noticing environmental cues. Without sufficient cues, humans cannot create an accurate or useful configurational schema. Relying solely on a configurational schema is therefore often insufficient for navigation, and humans, therefore, must utilize tools to aid the process. Perhaps the most widely adopted pedestrian class of navigation tools are 2D interfaces that rely on GPS. Not only is GPS cost-effective, but it provides real-time data on where the user is positioned so that the user does not have to rely solely on a configurational schema. Unlike paper maps, digital interactive map displays (like common car GPS devices or mobile map applications) rotates the map frame of reference to the user's world perspective, removing the cognitively demanding task of matching these frames of reference(Thorndyke and Hayes-Roth, 1982). It better focuses the user's attention on relevant aspects of a navigation task(Collins et al., 1978), such as faster vehicle driver response time(Srinivasan and Jovanis, 1997) which improves efficiency navigating to a destination(Lee and Cheng, 2008). Although digital interactive map displays are widely they fall short on certain aspects compared to traditional paper maps. Oulasvirta established that when users were given either a paper map or a digital device to navigate, users given a paper map developed a better mental model of the journey ahead of them whereas users given a digital device developed dependence on their device(Oulasvirta, 2005). In another study, users told to navigate an environment using a digital mobile map proved to be worse at estimating the route distance compared to users told to navigate with a paper map. This was credited to the fact

that “mobile map users acquire a more fragmented and regionalized knowledge representation based on strong connections between locally clustered landmarks along the route.”(Willis et al., 2009)

5.4.2 Direct Screen Display Based Navigation

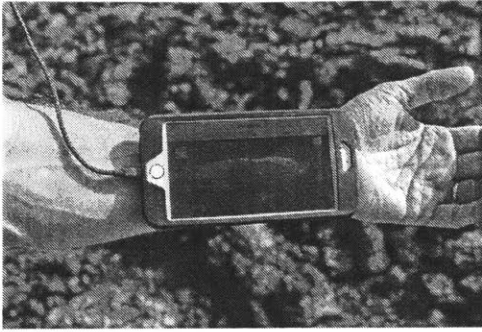
Although the most common screen based interface from past missions is a wrist display, we here further expand it to include hand-held solutions such as tablets, as both of these were tested during the BASALT campaign.

Wrist display have been used in several past analog missions, and proposed as part of the astronaut hardware. It can be used as an interface for displaying written messages from IV, an outline of tasks to execute, help keep track of the time line, and display a digital map to aid in situational awareness of mission targets, and general geospatial information. Within this work we focus specifically on the application for navigation.

Hardware Description

Due to the lack of environmental requirements in the context of BASALT, the function of a wrist display was emulated through the use of smart phones, enabling out of the box development of software for these phones. The phones selected where iPhones, which were used with an off-the-shelf wrist accessory to attach to the wrist, as depicted in Figure 5-4a.

This wrist display was used as an integral part of the simulations, however, a second display, which was not used by the EV crew, was also tested, a tablet, which was carried with a harness, as depicted in Figure 5-4b. This solution was originally envisioned to have a larger display to enable a user interface for SEXTANT that would be more interactive and easier to use in the field, than the small touch screen featured on the mobile display.



(a) Wrist display
Photo rights: K. Beaton



(b) Portable tablet
Photo rights: Andrew R. Hara Photography

Figure 5-4: Direct Screen Display integration during BASALT
Pictures from BASALT 2016 campaign, also used during 2017 campaign.

Functionalities

The navigation aid used for the wrist display was originally based of an off the shelf application, Google Earth. However, for our purposes, the Google Earth application had several shortcomings: it would require network to buffer map imagery, a task which could lead to frozen maps if the network dropped, thus removing any navigation capabilities. It would also require frequent manual re-alignment of the orientation of the map to align with the orientation of the user. Google Earth being a standalone software, it would have been hard to integrate it with SEXTANT, and add interface elements enabling capabilities such as re-planning. Therefore a custom interface was developed with these key capabilities: during its use at BASALT 2017 campaign(it's first use) it was named SEXTANT-WebApp, due to its web-based user interface. This made it platform independent, only requiring the user to have access to a web browser. The design and implementation of the interface was a joint development across a team at NASA Ames, at Cornell University, and here at MIT, and the interface is currently being upgraded further as an effort led by NASA Ames.

User Interface Design

Figure 5-5 shows the interface design as it was used during the 2017 Hawaii deployment. This is the view from a computer browser, a similar view adapted to the smaller

screen size would have been seen by the EV crew on the wrist display as shown in Figure 5-4a. The interface is split into two areas: a button interaction area, where the user can interact with the planner, and the map area. The map display shows the name of the plan on the lower bottom left of the image.

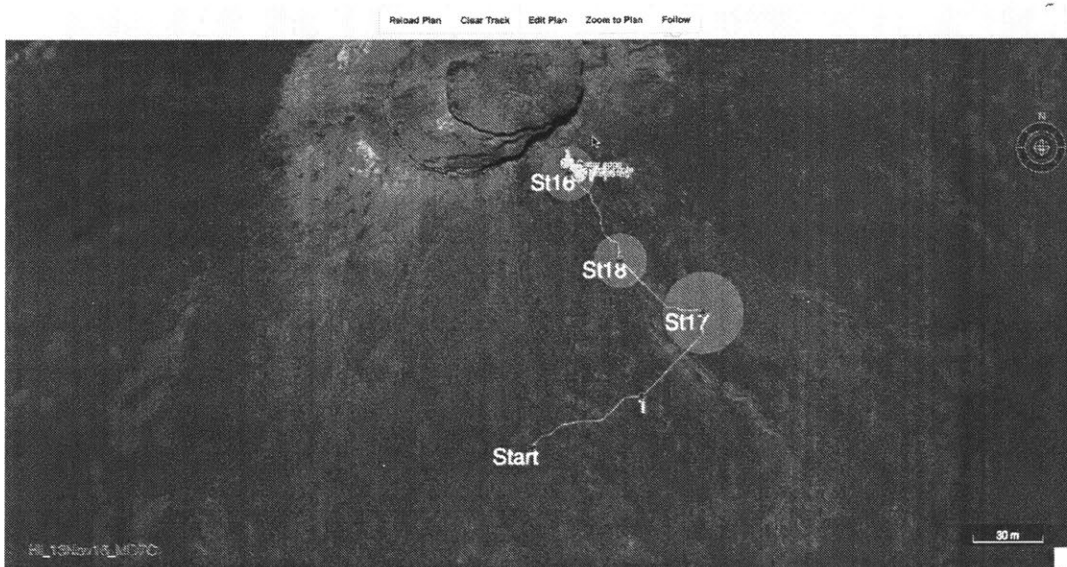


Figure 5-5: SEXTANT WebApp design

The interface comes with a drag and drop feature for the waypoints, and the re-planning can be done automatically when these are changed. The app also is used to display GPS position from the EV crew, their orientation while they walk, and general situational awareness info from the missions, such as messages that get "dropped" into the map from the IV crew.

The interface still relies on having a plan that was created with xGDS, which stores information on the more advanced options that can be turned on while using SEXTANT, and does not provide an interface to edit these, or to change the underlying DEM for a certain area.

5.4.3 Augmented Reality Display

In both real and simulated EVA missions so far the main communication line between the astronaut and the rest of the world (either Mission Control, or the IV crew, in the

case of an analog mission) has been through voice: the astronaut hears information from Mission Control or the IV crew, and can similarly relay information over a voice channel back to the team. In a similar way, so far any proposed interaction with automated support tools have proposed voice command as the main interface (I think I have some references on this).

The idea developed for navigation was that of a "virtual" path overlaid on the terrain within the Augmented Reality headset; this would serve as a virtual trail that the EV crew could follow, as outlined by the concept presented in Figure 5-6. The project got the name Holo-SEXTANT. This work was eventually successfully implemented, but involved a large set of challenges, due to constraints on the hardware, but also since this application was a very non-conventional application for Augmented Reality, as it has mainly been used in indoors environments. The next sections go more into detail on these limitation when we describe the hardware, and outline the technical solution used.

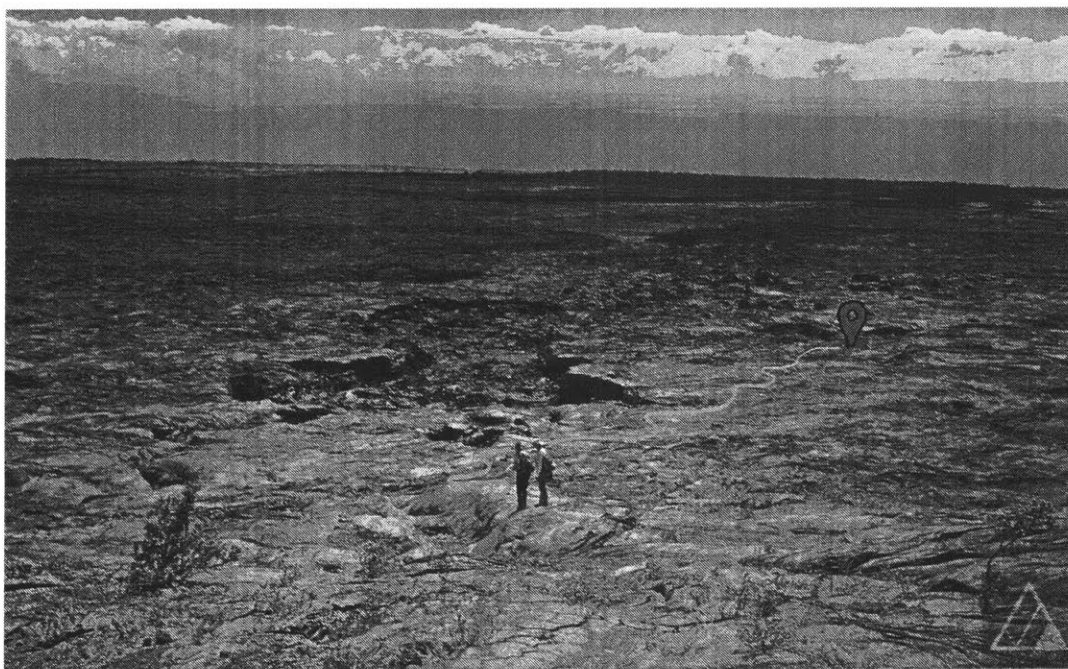


Figure 5-6: Vision of Holo-SEXTANT. Photo rights of original picture: Andrew R. Hara Photography

Hardware description

Our overall system consisted of a combination of Augmented Reality and a GPS system, in addition to various added systems for the AR solution to work in the outdoors environment, depicted in figure 5-7.



Figure 5-7: Hardware components used at Hawaii

The augmented reality capability was delivered through a commercially off the shelf available product: the Microsoft HoloLens, depicted in Figure 5-8. HoloLens is an AR smart-glasses Head Mounted Display (HMD), or headset with 6 degrees of freedom (DOF), an estimated 30°x 17.5°field of view (FOV), stereoscopic display with 1268 x 720-pixel resolution per eye, spatialized audio technology, WiFi and Bluetooth wireless connectivity. The HMD has an onboard computer consisting of a general-purpose processor as well as a custom Holographic Processing Unit (HPU). HoloLens has gaze, gesture, and voice (GGV) interface commands. The headset has a battery lifetime of 2-4 hours depending on the resource demand by programs running on it. The headset comes with a combination of cameras on-board, four environment understanding cameras, one depth camera, and one HD 2MP video camera which build up a three dimensional mesh of the surroundings, as these get explored. The meshing capabilities only reach about a 5m radius around the user, but as the user moves in his environment the mesh keeps building itself with the new sensory data. The HD camera can also be used for recording videos and taking pictures from the first person view, and project the holograms on top of the video or the picture, recreating

the view from the user, which is how the pictures from the HoloLens depicted later were created.

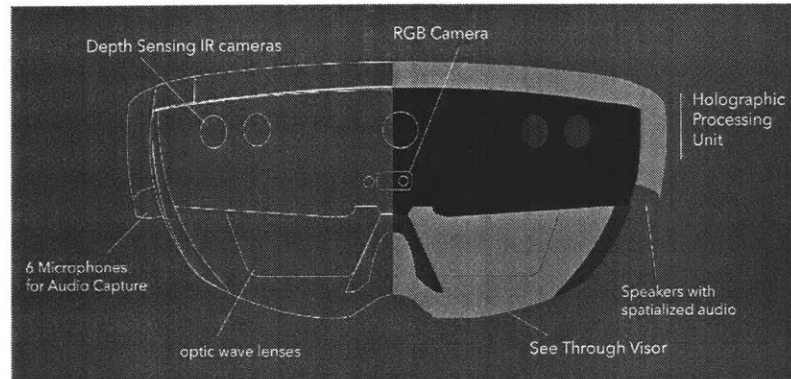


Figure 5-8: Microsoft HoloLens Headset

The headset has two lenses on which holograms are projected stereoscopically, which then creates the illusion of a three dimensional object in front of the viewer. Thanks to the meshing capabilities, the HoloLens allows for occultation of holographic objects behind real objects, which adds a level of realism to the experience. Creating images uses so-called "light addition" by which holographic objects are visible because of the light projected on the lens. This stands in contrast to a light removal lens, where the lenses would be opaque(light is removed) in ares of holographic objects, and transparent over the rest of the display. The choice for the first technique is clearly from a technology feasibility perspective. However it creates certain limitations on the HoloLens, as holograms will fade in highly lit environments.

Lighting is one of the challenges faced by using the HoloLens in an outdoors setting, such as that of path navigation. The headset was developed mainly for indoor use, and therefore a lot of environmental constraints from the outdoor were not taken into account. The factors we investigated, and tried to counter whenever needed, where first operating conditions that could be harmful for the HoloLens, and second environmental factor that could result in poor sensing capabilities of the HoloLens leading to either poor stability of holograms placed in the real world, or poor capture of gestures. Both 20% and 50% Visible Light Transmission (VLT) polarization films were tested. Intense sun at Kilauea required use of the darker 20% VLT film. With

the film applied, holograms were significantly more visible. Additional hardware modifications to HoloLens included an umbrella hat and a solar fan. Both of these were attempts at dealing with HoloLens' sensitivity to high temperature, heat, and sunlight. An umbrella hat was used to shield the headset from the sun's heat and sunlight. A clip-on solar fan was also added to the front of the umbrella hat to help cool HoloLens in the event of over-heating. When used in direct sunlight, the battery and other internal components of HoloLens heated up requiring this modification. A high-amperage portable power bank was also utilized for extending the battery-life of HoloLens, especially when used in higher than normal temperatures. To account for the high wind at the open landscape of Kilauea, an in-line microphone was used for higher accuracy voice commands.

The GlobalSat BT-821C high accuracy and high performance Bluetooth GPS receiver was used because the HoloLens does not have any GPS receiving capabilities built in. This model has a 24 hr battery life, 2.0 M accuracy, as well as 35 second signal acquisition time making it durable, accurate, and fast for Holo-SEXTANT's geolocation awareness. Aside from the modified HoloLens, a rugged industrial-grade tablet with the SEXTANT web-app was used to guide testing and test the accuracy of the Holo-SEXTANT application.

Operational conditions that were considered were exposure of the headset to heat radiation from the sun, risking over-heating, and effects of rain and wind. The first challenge was faced during testing outdoors in the beginning of the summer during a sunny day: the HoloLens would abruptly shut-off, and appear to be out of battery. As we later learned, this was due to the internal temperature of the HoloLens rising above a certain safety threshold, and turning itself off as a response. From previous years, we were aware of the possibility of rainy, and potentially windy, conditions during the Hawaii test. The wind would strongly affect the ability to do voice commands, and one solution which we implemented was connecting a pair of headphones with built-in microphone to get a better signal. For the rain, we brought both a regular umbrella, and a head-mounted umbrella, both which proved to be more of an obstacle than a help, since often time the rain was coupled with the wind, and it would be hard

to prevent the umbrellas from losing their structure. On the other hand, we learned that the HoloLens is much more robust to water than previously anticipated. During Mission Day 8 there was a constant misty rainfall during the entire operation, without any problems in the hardware.

On the environmental factors end we expected the lighting to be a problem because it might reflect too much off the rock and confuse the Infrared(IR) sensors used for meshing. As it turns out, this was not of a big concern after all, and the HoloLens did a really good job mapping its surroundings regardless of the weather conditions(from sunny to rainy). The only concern for the meshing was making sure the cameras did not get covered by water, which would diffract, or even sometimes block the light and confuse the sensors. The only real challenge we had to face was that of a light environment which made any hologram projected by the HoloLens seem very dim and transparent. This was solved adding a polarizing filter. Two filters were tried, a 60 percent and 80 percent; the latter proved to work better for our application.

Functionalities

HoloSextant was developed with simplicity in mind: focusing on the bare bones functionalities that could enable a basic navigation capability.

The main capability is visualizing the path for the EV crew to follow. The visualization of the path was the most simple possible: a straight line. Additional functionality was added to allow for debugging and a navigation menu was implemented to

System Data Flow

Holo-SEXTANT receives its data from SEXTANT. However, since this first prototype was offline and locally run, the paths were preloaded. A comma separated value(CSV) file with the latitude, longitude, and altitude data is generated and exported from the web app. This path is then uploaded to HoloLens. As many paths as desired can be uploaded to a pre-designated folder. Files can also be added at runtime on the field through a computer wirelessly uploading files to HoloLens. The bluetooth

GPS receiver connects with the Holo-SEXTANT app immediately after starting the application and sends GPS data constantly to Holo-SEXTANT providing it with geo-location awareness. With the path loaded and the GPS information streaming, Holo-SEXTANT can locate the user and render the holographic path relative to the user after some rotational calibration.

Calibration procedure

Holo-SEXTANT runs the Unity 3D game engine under the hood to visualize and render on top of the real world environment. The application was built using the Unity framework and C# as the scripting language along with the open source Mixed Reality Toolkit. The Mixed Reality Toolkit provides higher level HoloLens features and functions useful for development. This section will detail some of the technical implementation methods and challenges. Bluetooth communication is a foundational requirement for Holo-SEXTANT to receive GPS data. This was custom built for HoloLens to interface with the BT-821C receiver. In the background, a multi-threaded process searches, connects, and receives GPS data from the BT-821C. After parsing the NMEA 0183 GPS formatting, Holo-SEXTANT uses the data to update the current position of the user.

However, after parsing the data, there is still a coordinate system transformation needed. The solution path from SEXTANT is using local coordinates in the reference frame, defined by an origin and a set of axes, of the DEM used to generate the path. HoloLens uses a coordinate system using the headset start position as the origin (X to the right, Y upwards, Z towards the user). The path and the Bluetooth receiver both use GPS as a global coordinate system. Holo-SEXTANT takes the initial GPS coordinate of the user and uses a transformation matrix to convert all of the GPS points at runtime to HoloLens coordinates. Since the path and many other components are world-locked content, these GPS points are converted to distance measurements to place in HoloLens world. After identifying the transform, the path can be generated by transforming all of the coordinates from the SEXTANT generated path into relative distances and initialized where needed. After converting to

distance measurements, translational calibration of the path is complete, rotational calibration is still needed. Sensors in HoloLens ensure the calibration of the z, and x-axis rotations, however, the y-axis transformation is needed to properly place the path. An automated process was investigated using a series of GPS coordinates to estimate bearings of the user and rotate accordingly. However, a manual method proved usable and was used. The user identifies the start of the path and rotates the path until the path start and the actual start of the path align.

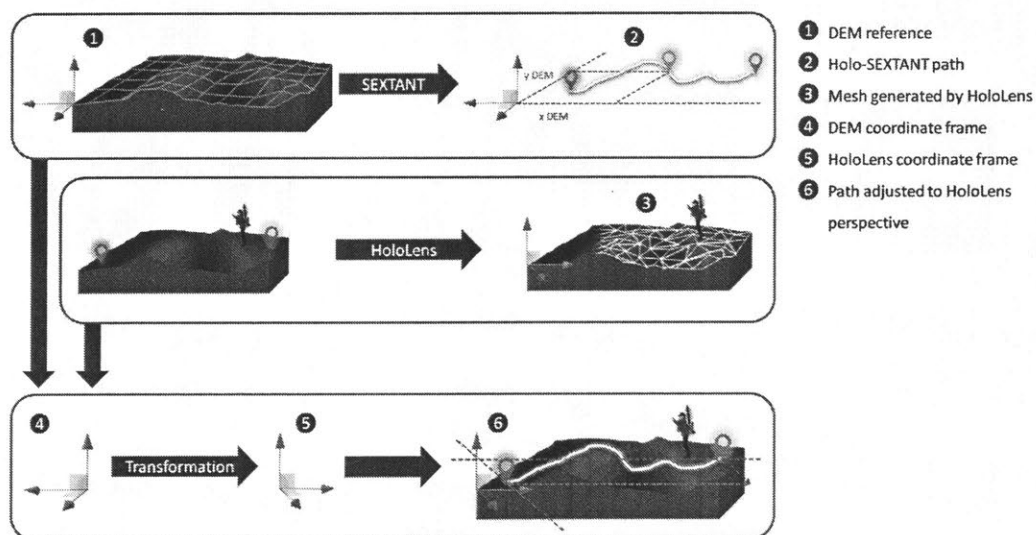


Figure 5-9: HoloLens Reference Frame Transformations

User Interface Design

When designing a user interface for navigation especially in treacherous terrain, several aspects have to be kept in mind: the user, the user's cognitive load, intended interactions, and the user's surroundings. By keeping these guidelines in mind, potential harm, and additional complications can be avoided. This section provides some of these guiding principles behind Holo-SEXTANT. First and foremost, the interface should not distract from the environment, since the environment is dangerous and simple distractions can cause slips, falls, and injuries. The UI components should not obstruct their field of view, or distract them with animations. All of the components

should be simple, easily visible, and intuitive. Since the interface is aiding the user to perform their task, it must play a supplementary role. The user cannot direct full attention to the application since they will also be walking, working with team members, and performing tasks. To prevent from distracting the user, Holo-SEXTANT has a simple, bright, and straightforward interface. Holo-SEXTANT consists of a 2D screen locked user interface and a 3D UI (holographic elements). The 2D UI is layered closest to the user so that the 3D objects in the user's view do not interpenetrate each other, and the menu displays are a translucent dark green and take up minimal screen space. All of the text is bright white for optimal visibility.



Figure 5-10: First person view of Holo-SEXTANT UI
Includes path, waypoint, and status display

The 2D static display is overlaid on top of all other elements, virtual and physical. It includes the primary information needed for the user. A small circular indicator shows the current GPS status, in three states: red when the BT-821C is not connected, orange when it is connected without satellite uplink, and green when connected and receiving. A direction indicator shows what direction the user is facing. This feature uses a GPS bearing estimation that works based on the starting bearing of the user. An implementation of rotational calibration to receive true bearings is under progress to show the true bearings of the user. On the right side of this heads-up display is

an overall status indicator showing whether Holo-SEXTANT is calibrated, showing the path, or not initialized. This heads-up display follows the gaze of the user as head-locked content. To reduce the visual clutter in the 2D static display, several extra features were removed such as time, and current GPS location; Only the most important information was kept.

The 3D UI consists primarily of a holographic path that follows the route described by SEXTANT generated path. The path is rendered bright blue to ensure maximum visibility under the harsh lighting conditions Holo-SEXTANT must operate in. It has simple geometries and therefore low visual fidelity, which allows for better spatial awareness of the 3D object and reduces the cognitive load required to visually scan the environment to locate the path. Omitting visual complexities (eg. texture, specularities) in the holographic path encourages the user to focus on the position of the path without being distracted by such irrelevant details. To make navigation even easier, several distinctive waypoints were added to the path at varying locations. These allow the user to deviate from the path if needed yet be able to identify the waypoints and walk towards them and continue following the path.

Perspective and occlusion are the strongest depth cues when manipulating and observing objects beyond an arm's length. HoloLens handles perspective view but is not suited for handling occlusion for objects far away. For instance, the end of a path that curves around a hill should be hidden to the user, but HoloLens does not know that there is such a hill because its depth-sensing cameras don't have a long enough range (unless it has been pre-emptively recorded and placed in the Unity environment), and therefore the part of the path that should intuitively be occluded the hill is still visible to the user. The holographic path does appear relatively thinner the further away it is from HoloLens, allowing the user to judge distance, but unless the user is concerned about gauging the route ahead, the user is almost always more immediately concerned with what is right in front of them. With an always present overlaid path, no recall or memorization is required, and the cognitive load is reserved for navigation and motor control through a potentially hazardous terrain. Aside from the heads up display and path, there is also a 3D status display. The status display is

locked to the user's view. This means that while not in the field of view of the user, it will always follow along with them wherever they walk. The display is located to the right of the user, such that they can always look over 90 degree to the right and see the display. This status display shows less critical information than compared to the HUD. It shows the actual coordinate value, velocity, distance to start of the path, and distance travelled total. A help display can also be brought up if needed to give a reminder of the voice commands and interactions possible. Both 3D menu displays are located at a comfortable distance from the user and never take up the entire field of view. Linear interpolation and slight delays are used for controlling the displays movement as the user moves without being jarring or disorienting.

Field Use Technology Demonstration

The testing of Holo-SEXTANT during the Hawaii 2017 deployment was mostly focusing on baseline functionalities and showcasing it as a proof of concept. As it was uncertain if the capability would be ready on time, the focus was put on getting the demo ready, and not on developing testing procedures. Hence no rigorous test plan was put in place, and this subsection describes simply the set of environments Holo-SEXTANT was tested in, experiences learned, and qualitative feedback from people who tried it out.

This section will elaborate on the user experience through the entire application. First, the user needs to pre-load paths obtained from SEXTANT web-app prior to the mission. The user can then start the application on the field as well as the BT-821C module. Once the user sees the green light indicating satellite uplink, Holo-SEXTANT is ready. The voice command- show path will generate the path and place it the appropriate distance away from the user. Now they can perform the rotational calibration by verbally commanding the system Rotate (-)[0.5, 1, 5, or 10] degrees to respectively rotate the path about the user's origin point. They can rotate until the path aligns with the correct start GPS coordinate. The user heads to the start point. The status display indicates the distance left to travel to start.

After arriving at the start point, the user can raise or lower the path as needed



Figure 5-11: EV crew navigating with Holo-SEXTANT during Mission Day 9 at Hawaii 2017 deployment

to fit their comfort level. All that is left is to follow the path to the destination. If needed, the user can raise, lower, rotate, and re-calibrate the path at any time during the traverse. They can always refer to the status display to see their bearing, GPS, distance traveled, and distance from start.

The functionalities were evaluated in three different settings: near the lodging facilities at Kilauea Military Camp in a very simple and flat environment, in the field outside of simulation, and in the field during simulation for the approach phase.

The functionalities were evaluated by a range of people involved in BASALT: engineers, scientist and students, but most importantly the crew itself, who received adequate training in how to interact with the interface. As Augmented Reality also constituted a new technology for many, first a general HoloLens training was carried out to familiarize people ahead.

Due to the novelty of AR interfaces, the Holo-SEXTANT application went through a stage of training to familiarize the crew, followed by field testing. Three full field tests were executed with EVA crew members unexposed to AR. One day was tested after the analog mission objectives were complete for the day. The other 2 field tests

were done during the mission for the 20-minute approach path into the region of interest for that day's field-work. A top view of the path taken as well as the path plotted can be seen in Figure 5-12 showing the use of Holo-SEXTANT to follow the pre-generated path.

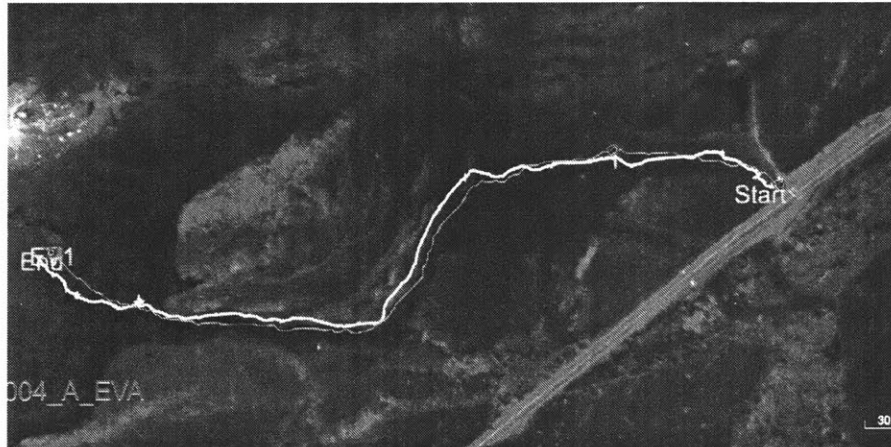
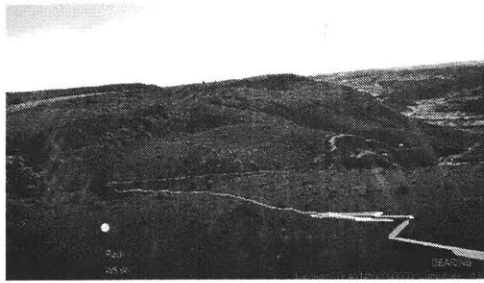


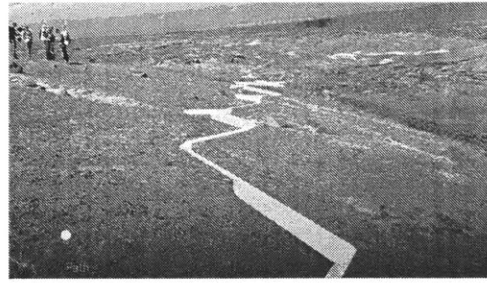
Figure 5-12: SEXTANT WebApp view of track generated by navigating with the HoloLens. In orange: planned path, in yellow GPS tracks from Field Test Day 1.

Field Results and Lessons Learned

After implementation and arriving on the field, several of our assumptions were validated and some were invalidated. The efficiency of our calibration methods, preparation for environmental conditions, and accuracy of the path were all segments that we learned significantly of from the deployment. To start with the calibration, we found the GPS receiver to have accuracy problems at times. While the advertised 2.0 m accuracy was exhibited most of the time, there were some times when the GPS would be inaccurate. This would lead to incorrect calibration. Discounting this occasional hardware issue, our translational calibration was very accurate providing 1-2 m accuracy. Our automated rotational calibration mechanism was not accurate enough to test or use on the field. Manual calibration, however, was efficient enough to not be cumbersome. With less than 5 voice commands issued, the path can be calibrated completely. Sometimes, however, recalibration was needed during the middle of the path, if the path required precise traversal. For this reason, a tablet was running the



(a) Out-of-sim testing



(b) In-sim testing

Figure 5-13: Screen capture from within the HoloLens during Hawaii 2017 testing

SEXTANT web app showing the path on the terrain, allowing the user to re-calibrate if needed if it was not in the same position as the tablet visuals.

Performance of HoloLens in the outdoors was surprisingly robust. Anticipated fragility due to rain was not a problem, as was discovered during the second in-mission field test. HoloLens operated continuously in a mixture of windy conditions with fluctuating rain, as long as the screen was dried off to clear the sensors and the view of the crew. The head umbrella was not very effective in windy conditions, as it would be hard to keep it in position, making the inherent robustness to rain a significant advantage. Sun and heat also proved to be less of a challenge than expected; IR reflection on the rocky terrain only seemed to confuse the HoloLens sensors on rare occasions, and the mesh accurately captured the surroundings when visualized. As a result, holograms were stable during most of the tests. However, we did notice that if HoloLens was pointed to directly at the sun at any point it would either shut down or lose track of the surroundings. We believe this would be due to an overexposure of the IR sensors for enough time for HoloLens to lose track of its orientation and position in the current mesh, and as a result, losing the capability to extend the mesh further. Although the head umbrella proved of little use in the rainy conditions, often because of the winds, it was useful in sunny conditions as it would serve as a visor to keep the HoloLens IR sensors from being directly exposed. We also observed that rebooting HoloLens could often lead to an incorrect interpretation of the location of the user, as the path would no longer be in the same location and orientation, indicating an incorrect load of the previously existing

mesh. Visibility of the holograms was clear, thanks to the darker lighting offered by the tinted layer applied on HoloLens. In artificial lighting conditions, the tint was even enough to make the holograms occult the surroundings. This made for a clear visual of the path even in sunny conditions, while offering the tinted functionality of polarized sunglasses, which the crew would have required anyways. One larger challenge however, which had not been initially foreseen, was using voice commands in wind; HoloLens would require several attempts at the same command, and the commands would sometimes have to be communicated with a very loud voice. This would be frustrating and distracting from a user perspective, but we had a simple fix: connecting a custom microphone to HoloLens and using sound mufflers to reduce environmental noise.

Discussion

Despite several key assumptions and limitations in the hardware, HoloLens has sufficient hardware capabilities for a proof of concept navigation assistant. Some of the limitations did hinder the EVA experience at times. Even after initial calibration, drift, and other factors can lead to a need for recalibration. In order to re-calibrate, the EVA crew member would have to stop, assess where they are and rotate or translate before proceeding. One method of increasing accuracy and preventing such recalibration is a dynamic updating and calibration mechanism. Firstly, an automated rotational calibration method is needed. Second, with the GPS data stream, the application should assess the rendered path accuracy at some nominal update frequency to ensure that the path remains calibrated. While design ideas were outlined, such dynamic updating functionality has not yet been implemented. Automatic rotational calibration was one of the challenges that we faced. Rotational calibration requires some sort of compass information. One method that we explored was using a straight line of GPS points recorded at initialization to estimate the true bearings of the user. Based on this, the path can be initially rotated in the right orientation. However, this proved quite difficult to implement, and also is not a robust enough method to perform re-calibration during a mission. It requires user input making it non-ideal.

Integrating a digital compass is another option, but this poses its own challenges with accuracy. A third more novel approach could be a larger architectural change for future research. One could mimic the spatial mapping technology that HoloLens uses to localize itself indoors. Aside from using WiFi, HoloLens generates a sparse map of its immediate surroundings and compares that with a global mesh that it has stored to find a match and locate itself in the global mesh. Mimicking this method, an external LiDaR or depth sensing camera alongside a wearable computer could be used to generate dense 3D maps and match them with the bigger DEM that was already acquired. Using GPS to localize on the terrain, and using a dense 3D map to identify the user orientation in the terrain, can lead to an efficient and automated calibration mechanism. A dense 3D map will likely be required because a sparse 3D map would not be able to capture enough features of the terrain. This method would require additional hardware for capturing as well as processing this information. However, we think that exploring methods to automate calibration and localization are crucial for future research to make an Augmented Reality navigational application robust. Some other challenges were errors in conversion between systems. At each stage of the data flow, the error propagates and accumulates. The initial DEM used to generate the SEXTANT path has an inherent error. SEXTANT has some tolerance for error as well, particularly because minute terrain features cannot be captured in SEXTANT. When this path is converted to the Unity coordinate system, there is more error introduced. However, for an EVA mission, the region of interest is typically larger in radius than a specific, precise, GPS coordinate. This allows for Holo-SEXTANT or any other EVA navigational tool to tolerate such errors. Despite this, some terrain and pathways will require much more precision. In our field test, we found one location with only a single naturally formed channel to be traversable. And there was a single accessible entry point into this channel. Because of which, Holo-SEXTANT had to be quite precisely calibrated to not lead the user astray drastically. Through the field testing at Hawaii Volcanoes National park, we found it hard to identify key metrics to test the efficacy of a navigation interface. Paths are not repeatable by the same user because the user now has a memory of traversing it. The speed of

crew members traversing had quite a large spread. Comparing the planned path with the traversed path might seem to be an obvious choice, however following the path exactly is unfeasible and unnecessary to the goal of the EVA. Furthermore, one of the goals of Holo-SEXTANT is to enable EVA crew to safely and willfully deviate from the path to explore if they wish to and return to the path with ease. Further investigation is needed to identify the best metrics to test such an AR navigation interface.

From user interviews and feedback from users, a couple of key points were highlighted. Visibility was a key feature that users liked. While the holograms were initially difficult to see, our modifications with the polarization, improved it significantly. One EVA crew member mentioned that wrist displays are very hard to see in the sunlight, making Holo-SEXTANT significantly better in visibility. Another insight was the customizability needed for different users. At a high level, different EVA crew members have different roles, and one single interface won't fit all of their needs. For example, the lead crew member might navigate for the entire team, while the second crew member is in charge of other operations. While they could both benefit from an Augmented Reality information display, their needs are quite different. For this reason, we made a highly customizable UI. Any UI component can be hidden if the user wishes so. All of the UI components were directly manipulatable allowing the user to design their own "workspace" on the go.

From walking alongside several users using Holo-SEXTANT, we noticed different user application interactions. One user raised the path to render above their head such that they would look up occasionally to gauge his progress and continue onwards independently. Others would place the path slightly above terrain level to be able to see the ground and the path simultaneously. Some users found it uncomfortable to see the world through a polarized HMD especially when the terrain can crack and watching each step is crucial. They would raise HoloLens to walk and lower it onto their eyes occasionally to check the path. Ultimately, we discovered that the interface has to adapt to the user for navigation and was designed as such. However there are other interface components that we thought of, explored and did not integrate. One

user option that should be explored in future research is different path visualization techniques. Bread crumbs, waypoints, arrows, and solid lines are all different methods possible. We found that having a very discrete path with several kinks and turns might not be the best method since humans do not walk with that level of precision.

Looking towards for future

During the development of Holo-SEXTANT we encountered a set of items that could improve the system, but which we didn't have the opportunity to address, either because of technological constraints or going beyond the scope of the original goal.

- *Alternative path visualization.* Holo-SEXTANT makes use of a long continuous holographic path to indicate the route described by SEXTANT, but there are alternative ways of displaying the same information. For instance, placing waypoints in the terrain, using arrows to point towards the next waypoint, making more use of colour to convey information.
- *Increased field of vision (FOV).* Navigating with the restricted horizontal FOV on HoloLens increases the angle and frequency at which the user must rotate their head to identify and follow the path. A restricted horizontal FOV (30° on HoloLens compared to 180° on humans) can have a non-trivial effect on task performance. A low FOV hinders performance at visual scanning tasks, and can affect distance estimation, speed and accuracy when manoeuvring through a physical obstacle course. Additionally, a wider FOV will have the added benefit of allowing the user to look ahead and anticipate upcoming parts of the environment. Although the current available AR HMDs do not allow for a sufficiently large FOV, we hope that advances in AR hardware development will make way for HMDs with a larger horizontal FOV. Measuring biometrics. SEXTANT was designed to calculate the shortest, safest, and most energy efficient route between points on a terrain. Monitoring EVA biometrics (such as heart rate, calories burnt) to display to the user could allow the user to be more aware of how different phases of their navigation in an EVA mission is affecting their

body.

- *Measuring the user's invisible states.* Being able to monitor invisible metrics such as intention could provide useful data for planning navigation routes. HoloLens does not currently support eye tracking capabilities but tracking user gaze could build a better picture of what the user is focusing on during an EVA mission.
- *Displaying distances and current position.* Estimating distance without additional tools is challenging for humans, especially in a terrain with sparse landmarks like Mars. In a study where users travelled by following either a paper map or a mobile map interface, subjects from both groups underestimated their travel distance (mean of 98.93 m using a paper map, 135.90 m using a mobile interface)¹¹. It is important that the user has a clear idea of what to expect for the terrain ahead of them. This can be accomplished by providing a 2D representation (like a mini map) or a 3D representation (like a hologram of the terrain). Additionally, by including a planning pre-task where a schematic overview of the entire route is displayed and can be referred to during the task, the user not only develops a better understanding of what to expect, but it also addresses the issue of overcoming an unstable configurational schema.
- *Reviewing completed traversals.* Holo-SEXTANT's capabilities end as soon as the user ends the EVA mission. A helpful capability would be to provide a 3D representation of the mission traversal for analysis, noting details such as biometrics, how closely the user stayed on the proposed path, changes in speed etc. This visualizes the data for a remote support team and could help the user mentally compartmentalize and contextualize segments of the EVA mission, which is especially useful if the same environment were to be traversed again. Similarly, pre-mission visualizations can be useful to plan the mission routes, and understand the terrain.
- *Adapting for flight-ready hardware.* As mentioned before, HoloLens is a general

purpose augmented reality HMD designed for indoor use. A next step would be to investigate custom hardware solutions using more powerful sensors and eliminating components that are not needed. Integrating with the space suit and conforming to the power and design requirements of NASA is another whole challenge in itself.

5.4.4 Conclusion

An augmented reality-based navigation system can readily be developed with off-the-shelf components. Although several methods, both digital and analog, exist for an EVA crew to follow their traverse plan, we believe that using AR methods could add significant value compared to current methods. It could improve situational awareness, provide insights on the path and the terrain other digital methods couldn't offer. AR is also an interface that could take full advantage of automated path planning as provided by SEXTANT. The current work partially fills in the gap that would allow an AR navigation interface to be compared to other methods such as digital display interfaces. Although we don't have any data on human performance using both of these methods we outline the feasibility of this method. We have outlined the setup that would allow future efforts to focus on the comparison of the AR platform with other interfaces. We also outlined the limitations we encountered in our setup and proposed ideas on how to improve this in future iterations of the current AR navigation architecture. These are key to make the system robust enough so that any comparison across information display methods is not negatively biased towards the AR solution due to flaws in the technology.

Ultimately we believe other EVA related capabilities could also be significantly enhanced from the use of AR. Enhancements could come from different stages relative to the EVA, and for a diverse set of stakeholders. Pre-EVA planning for scientist and the operation team, could benefit from tools like NASA's Jet Propulsion Lab (JPL) OnSight tool, which can virtually replicate the real environment of proposed EVA areas and immerse the user in this environment in a collaborative manner simultaneously with other users. The same tool could also be used in combination

with additional data post-EVA to allow a broader pool of researchers gain access to the mission. Mid-EVA applications could go beyond navigation, which was the main point explored in this paper, and both enhance communication across the intravehicular crew and the extravehicular crew. Such a capability could be provided by a tool similar to Skype for HoloLens. Despite the time delay, the science team in the mission support center could also benefit from an AR environment mid-EVA, being able to follow the EV-crew and their interaction with their environment from a virtual vantage point.

Chapter 6

Conclusion

This thesis outlined changes that were made to SEXTANT to adapt it for real use in future analog - or real planetary EVA missions. SEXTANT is easily available for integration with other planning systems, either through the Python interface described in the Appendix, or through a simple API that can be easily called over the server. The thesis focused to address the challenge of rough terrain, but made partial contributions towards measuring the performance of SEXTANT, and improving its performance. The thesis also presents work on different interfaces that have been designed to interact with SEXTANT, notably two that have been recently developed partially, or entirely, here at MIT: the SEXTANT Web App, that was developed as a collaboration effort across NASA Ames, Cornell and MIT, and the Holo-SEXTANT, which was pioneered by Anandapadmanaban et al. (2018).

The rough terrain is addressed through a slight expansion of the current path planning formulation; the thesis shows that this is an effective method, but results in slower performance in exchange. Finally, SEXTANTs application during BASALT is detailed, showing SEXTANTs applicability in a real world environment.

6.1 Future work

Significant improvement can still be obtained from path planning. One of the questions that previous work on SEXTANT has kept raising is the question of speeding

up the performance of SEXTANT to be able to do real-time planning. This is where the literature discussed in Chapter 2 should be further investigated to adapt better state of the art methods.

An interesting question that wasn't discussed as far as the author would have liked to is the question of resolution, and what resolution is required to capture details in the map relevant for path planning. Although it is hypothesized that the resolution required should be on the scale of a person being, there is currently no analysis justifying this.

While the question of resolution is discussed, another key metrics that could speed up the algorithm would be reducing the search space. The question would be how to reduce the map representation to a potential lower resolution while still capturing most of the features relevant to planning. Part of this is captured in triangulation method discussed in Chapter 3. Alternatively, the path planning algorithm might interpolate values from the surface, instead of considering only the discrete states, such as is done in A*.

Finally, there is a significant gap in the modeling fidelity. Currently the velocity model is based of estimates from lunar traverses; this should be updated to Earth based models which exist. Further works should also look into velocity and energy models for traversing in different terrains.

Appendix A

SEXTANT Python API

```
[1] import deps
    from pextant.lib.geoshapely import GeoPoint, \
        LAT_LONG, LONG_LAT, UTM, Cartesian, Cartesian2
```

Abstraction for Earth surface geometry

SEXTANT uses objects - whose names are prefixed by *Geo* - to represent geometrical objects that are defined on a map. The objects are built on top of the python library shapely, that is standard for generic geometry not attached to a map representation. Here we address the gap by providing a very basic abstraction that makes it easy to convert between different Earth based reference systems.

GeoPoint

GeoPoints take a coordinate system, and then two coordinates (currently not extended to 3D). The coordinate systems predefined are as follows:

- Latitude, longitude (LAT_LONG), or inversly longitude, latitude (LONG_LAT)
- UTM(Universal Transverse Mercator). These are locally projected cartesian grids that depend on which longitude you are at (defining a zone), and therefore take as an argument the zone number. Of the UTM grid. Optionally, it can also take a previously defined point as an argument to automatically determine the UTM zone.
- Cartesian(point, resolution). This creates a 2D cartesian coordinate system centered at a point, with grid spacing set by the resolution. Coordinates here represent cells, and are therefore integer
- Cartesian2(point, resolution). Same as Cartesian, but continuous. Should find a better name

Latitude, longitude

```
2. geopoint = GeoPoint(LAT_LONG, 47.9941214, 7.8509671)
```

These can next be converted to other coordinate systems; the output are the coordinates (and not a new object) - this might be a good upgrade to add. One caveat, if we convert to UTM of any other zone, it will still force the conversion to UTM of the zone that the point is in.

```
3. geopoint.to(UTM(5)) #Any number instead of 5 will give same result
```

```
array([ 414278.16731025,  5316285.59492359])
```

Cartesian

```
4. ref_frame = Cartesian(geopoint, resolution=1)
```

```
5. geopoint.to(ref_frame)
```

```
array([0, 0])
```

```
6. point_in_ref = GeoPoint(ref_frame, 20, 10)
   point_lat_long = point_in_ref.to(LAT_LONG)
```

And converting back

```
7. GeoPoint(LAT_LONG, *point_lat_long).to(ref_frame) # notice that the * sym
```

```
array([20, 10])
```

GeoPolygon

A collection(list) of several GeoPoints are defined as a GeoPolygon

```
[8] from pextant.lib.geoshapely import GeoPolygon
```

```
[9] geopolygon = GeoPolygon([geopoint, point_in_ref])
```

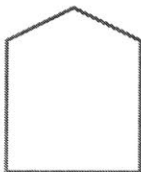
Since the library is built on top of shapely, we can also visualize the shapes natively in the notebook

```
[10] geopolygon
```



We can also define a polygon through arrays of coordinates in a coordinate system, in a similar way we defined the GeoPoint

```
[11] house = GeoPolygon(ref_frame, [0,0,10,20,20,0], [30,10,5,10,30,30])  
house
```



And easily convert to other systems in the same way that was done with GeoPoint

```
[12] house.to(LAT_LONG)
```

```
array([[ 47.99385154,  47.99403145,  47.99407776,  47.99403413,  
        47.99385422,  47.99385154],  
       [ 7.85097309,  7.85096911,  7.85110211,  7.85123713,  
        7.85124112,  7.85097309]])
```

One can access elements of the list through regular indexing:

```
[13] print(house[3].to(LAT_LONG))
      house[3].to(LAT_LONG) == house.to(LAT_LONG).T[3]
```

```
[ 47.99403413  7.85123713]
array([ True,  True], dtype=bool)
```

Helper functions

Geo shapely is built on top of shapely, where the underlying representation uses the UTM coordinates. This could result in small numerical inaccuracies due to the fact that UTM coordinates can be numbers on the order of a hundred thousands or a million. The nice thing is that we can use it in the same way you would use a shapely object:

```
[14] geopolygon.envelope.intersection(house)
```



```
[15] geopolygon.envelope.intersection(house.envelope)
```



GeoEnvelope

Although shapely offers the envelope function to get the envelope of a polygon, a very light object was developed in addition for specific use cases where it was useful. This representation makes it easy to add a buffer to the envelope by calling `addMargin(scale, length)`, and has a convenient function to get the upper left and lower right coordinate of the envelope. The envelope is defined through the upper left and lower right corner, but all GeoPolygon objects have the built in function `geoEnvelope()` to get the envelope.

```
[1] import deps
```

1. Abstractions for representing environments

Environmental models can be represented either through a GridMeshModel or a TriMeshModel, using a grid and a triangular based representation of the environment, respectively. Here we will document how to use the GridMeshModel representation. Several ways exist to define the environment, we will start with the simplest one, which is based on having an array of elevations where each entry to the array represents the elevation at a coordinate corresponding to the row and the column.

```
[2] from pexant.mesh.abstractmesh import NpDataset
import numpy as np
```

```
[3] xx,yy= np.mgrid[0:5,0:5]
```

```
[4] basic_terrain = NpDataset(0.1*(xx**2+yy**2), resolution=1)
basic_terrain
```

```
array([[ 0. ,  0.1,  0.4,  0.9,  1.6],
       [ 0.1,  0.2,  0.5,  1. ,  1.7],
       [ 0.4,  0.5,  0.8,  1.3,  2. ],
       [ 0.9,  1. ,  1.3,  1.8,  2.5],
       [ 1.6,  1.7,  2. ,  2.5,  3.2]])
```

This dataset is wrapped around numpy so we can access can easily access entries:

```
[5] basic_terrain[1,1]
```

```
0.20000000000000001
```

Or access several entries, and even interpolate

```
[6] basic_terrain.get_datapoint(np.array([1,1],[1.5,1.5]))
```

```
[16] from pextant.lib.geoshapely import GeoEnvelope
```

```
[17] geoenvelope = GeoEnvelope(house[0], house[3]).addMargin(5, 5)
```

```
[18] geoenvelope.envelope
```



```
[19] house_upper_left, house_lower_right = house.geoEnvelope().getBounds()
```

```
[20] house_upper_left.to(ref_frame)
```

```
array([0, 5])
```

```
[21] house_lower_right.to(ref_frame)
```

```
array([20, 30])
```

```
array([ 0.2,  0.5])
```

1.1 GridMesh

If we want to anchor the mesh to a geographical location, we will use the class `GridMesh`, and supply the coordinate of the upper left point of the terrain. Its important that this representation is agnostic of what the dataset contains; so far this just represents an abstract dataset with resolution, rows and columns (doesnt have to be a terrain)

```
[7] from pexant.EnvironmentalModel import GridMesh
    from pexant.lib.geoshapely import GeoPoint, LAT_LONG

[8] upper_left_corner = GeoPoint(LAT_LONG, 0, 0) # this will be the north-west
    basic_mesh = GridMesh(upper_left_corner, basic_terrain)
```

We can read out some basic properties of the mesh, and plot it

```
[9] print basic_mesh

height: 5
width: 5
resolution: 1
nw corner: POINT (166021.4430805405 0)
```

The upper left corner can be accessed, and returns our original anchoring point. The lower right corner is also accessible for convenience:

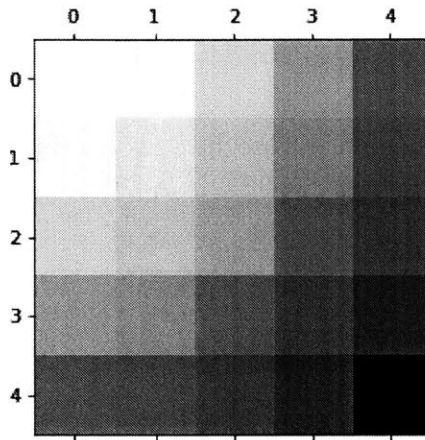
```
[10] upper_left_corner, lower_right_corner = basic_mesh.nw_geo_point, basic_me
```

We can also access the original terrain dataset through the `dataset` keyword

```
[11] import matplotlib.pyplot as plt115
```



```
plt.matshow(basic_mesh.dataset, cmap='gray_r')
plt.show()
```



GridMesh also stores the local coordinate system of the grid, which can then be converted back and forth to other representations. The two coordinate systems are called ROW_COL and COL_ROW, which allows to define a point given the row and the column.

```
[12] point_in_mesh = GeoPoint(basic_mesh.ROW_COL, 1, 1)
```

1.2 GridMeshModel

This is when we transform a terrain into a representation that the path planner can use. This adds a large set of methods to terrain useful for its analysis. The easiest way to generate the model is to just derive it from the GridMesh we already have. To do so, we can load a smaller part of the dataset we have so far, say for example in a given envelope.

All the methods from GridMesh are still available in GridMeshModel

```
[13] from pexant.lib.geoshapely import GeoEnvelope
```

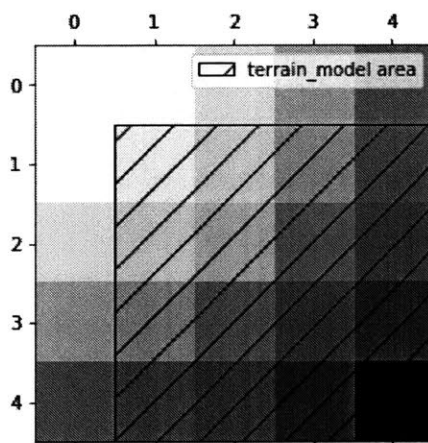
```
[14] model_envelope = GeoEnvelope(point_in_mesh, lower_right_corner)
terrain_model = basic_mesh.loadSubSection(model_envelope)
```

If no envelope is passed as an argument, the entire dataset is processed. Be careful with this, as it might take up significant memory if a very large dataset is being used.

```
[15] import matplotlib.patches as patches
```

Before we used dataset to access the underlying heightmap. For the model we will use the data property, which carries the raw representation of the data.

```
[16] plt.matshow(basic_mesh.data, cmap='gray_r')
plt.gca().add_patch(patches.Rectangle(point_in_mesh.to(basic_mesh.ROW_COL
plt.legend(["terrain_model area"])
plt.show()
```



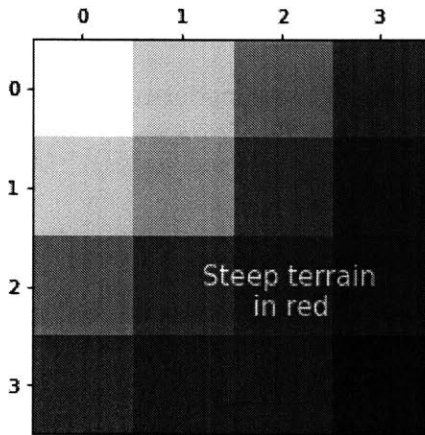
1.3 Model info

The model contains information on the slopes, and an obstacle map, which is currently set to terrain steeper than 35 degrees

```
[17] terrain_model.slopes
```

```
array([[ 22.98976777,  26.56505118,  33.85451481,  37.29207574],
       [ 26.56505118,  29.4962085 ,  35.79575991,  38.87665514],
       [ 33.85451481,  35.79575991,  40.31554221,  42.6746455 ],
       [ 37.29207574,  38.87665514,  42.6746455 ,  44.71062246]])
```

```
[18] plt.matshow(terrain_model.dataset, cmap='gray_r')
      obstacle_transparent = np.ma.masked_array(np.ones_like(terrain_model.data
      plt.imshow(obstacle_transparent, alpha=0.5, cmap='bwr_r')
      plt.text(1.2,2.3,"Steep terrain \n      in red", size=15, color="white")
      plt.show()
```



The code below demonstrates a more advanced usage of GeoMesh

2 Importing DEMs

2.1 GeoTiff

SEXTANT comes with handy helper objects to help import GeoTiffs, which are the preferred datatype for Data Elevation Maps (DEM). This is done with a library called GDAL, the Geospatial Data Abstraction Library, and hence the class is called GDALMesh. The GDALMesh class inherits from GridMesh, meaning that we can play around with it in the exact same way we did with the earlier example.

```
[19] from pextant.EnvironmentalModel import GDALMesh
```

We will use a 0.5 resolution DEM of NASA Ames Roverscape site

```
[20] ames_gridmesh = GDALMesh('Ames.tiff')
```

```
[21] print ames_gridmesh
```

```
height: 237  
width: 161  
resolution: 0.5  
nw corner: POINT (582678.314647603 4141911.40834981)
```

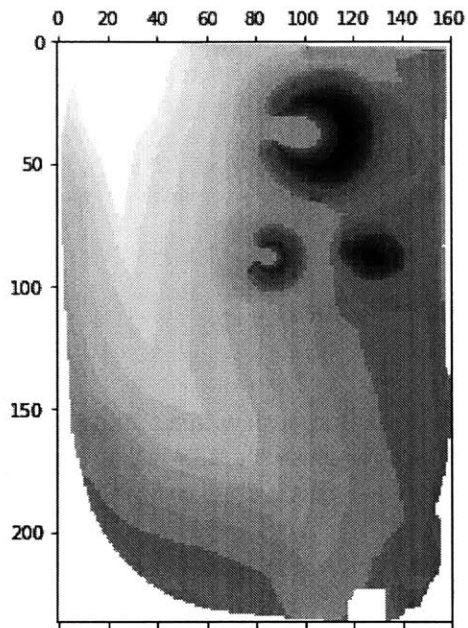
Whats different in this representation from when we had a GridMesh with a numpy array, is that if we access the dataset we wont get an array. This is because the DEM is still encoded, and wont be decoded until loadSubSection has been called; this is done to limit memory used when larger DEMs (100s of MB or GB size) are being used.

Since we see its a small dataset, let's just load it fully:

```
[22] ames_model = ames_gridmesh.loadSubSection()
```

We can display it, including the obstacles in red

```
[23] plt.matshow(ames_model.data, cmap='gray_r')  
obstacle_transparent = np.ma.masked_array(np.ones_like(ames_model.data),  
plt.imshow(obstacle_transparent, alpha=0.5, cmap='bwr_r')  
plt.show()
```

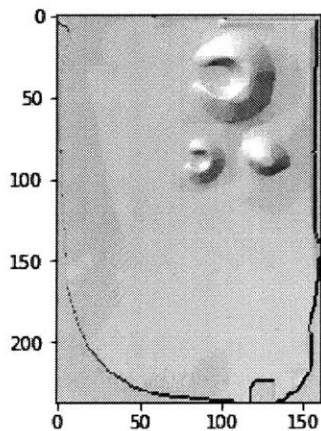


We notice that some of the areas are white; these represent masked locations, that are points with a no data value such as -9999.

Now that we are dealing with real data, let's also do a hillshade visualization in matplotlib:

```
[24] from pexant.viz.utils import hillshade
```

```
[37] hillshade(ames_model, 5) #5 is used to exaggerate the effect of the hills
plt.show()
```



2.2 From text file

Legacy terrain for SEXTANT was stored in text files, and there is a simpler helper function that can load it as a GridMeshModel(so dont need to loadSubSection)

```
[30] from pextant.EnvironmentalModel import load_legacy
```

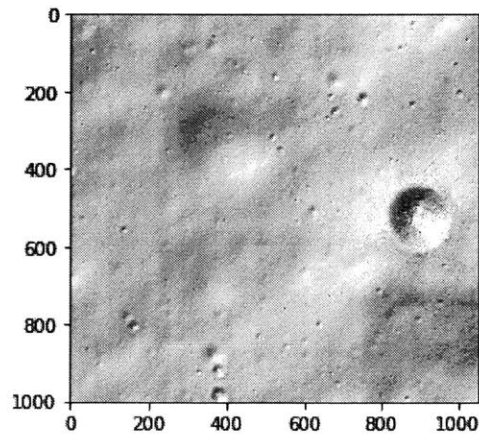
LOLA(Lunar Orbiter Laser Altimeter) instrument recently generated a 2m DEM of the Lunar terrain. The data has been post processed into the format of the legacy code, and is displayed below as an example.

```
[39] apollo14_model = load_legacy('Apollo14.txt')
```

```
[40] print(apollo14_model)
```

```
height: 1001  
width: 1051  
resolution: 2.0  
nw corner: POINT (0 0)
```

```
[34] hillshade(apollo14_model, 1)  
plt.show()
```



3) Other Representations: TriMesh

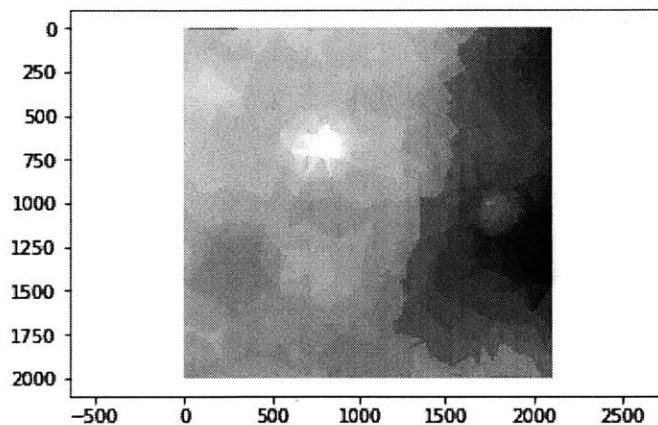
SEXTANT can also triangulate the terrain and represent it as a TriMesh. It triangulates the grid, and then uses an algorithm developed by Garland to decimate it and generate triangles in areas that need a larger density of triangles to accurately describe the terrain.

```
[42] from pexant.mesh.triangularmesh import grid_to_tri

[71] apollo14_tri = grid_to_tri(apollo14_model, accuracy=3)

[72] tri = apollo14_tri.data

[79] plt.gca().invert_yaxis()
plt.tripcolor(tri.vertices[:,0], tri.vertices[:,1], tri.faces, facecolors)
plt.axis('equal')
plt.show()
```



```
[1] import deps
```

Path Planning

1. Problem setup

The solver requires an environment and an agent(denominated explorer). The agent determines the cost function definitions.

```
[2] from pextant.explorers import Astronaut
    from pextant.EnvironmentalModel import load_legacy
```

To illustrate the use, our example will use the Apollo 14 map. Maximum slope is set to 15 degrees which is reasonable for astronaut traverses. To speed up the solver for later we need to pass in the argument `cached=True` to the `loadSubSection` method. This will carry out some preprocessing that will speed up solution time later on.

```
[3] apollo14_model = load_legacy("Apollo14.txt").loadSubSection(maxSlope=15,
    agent = Astronaut(80))
```

```
[4] import matplotlib.pyplot as plt
    import matplotlib.patches as mpatches
    from matplotlib_scalebar.scalebar import ScaleBar
    from pextant.viz.utils import hillshade
    from pextant.lib.geoshapely import GeoPoint, GeoPolygon
```

```
[5] start = GeoPoint(apollo14_model.ROW_COL, 750, 100)
    goal = GeoPoint(apollo14_model.ROW_COL, 650, 900)
    waypoints = GeoPolygon([start, goal])
```

We will choose for start the location of the Lunar lander, and for goal the border of the crater rim, this is to very roughly approximate the actual EVA - which has several intermediate points included.

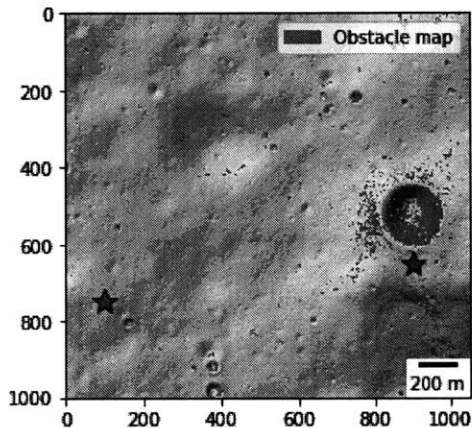
```
[6] hillshade(apollo14_model,1)
```



```

plt.imshow(apollo14_model.obstacle_mask(), alpha=0.5, cmap='bwr_r')
plt.plot(*waypoints.to(apollo14_model.COL_ROW), \
         linestyle='None', marker='*', markeredgecolor="k", markersize=15)
red_patch = mpatches.Patch(color='red', alpha=0.5, label='Obstacle map')
plt.legend(handles=[red_patch])
plt.gca().add_artist(ScaleBar(apollo14_model.resolution, location=4))
plt.show()

```



2. Solver setup

```
[7] from pexant.solvers.astarMesh import astarSolver
```

SEXTANT had a slightly unefficient implementation, so this was just changed at the tailgate of the project. It implements a slightly modified version of networkx's astar solver.

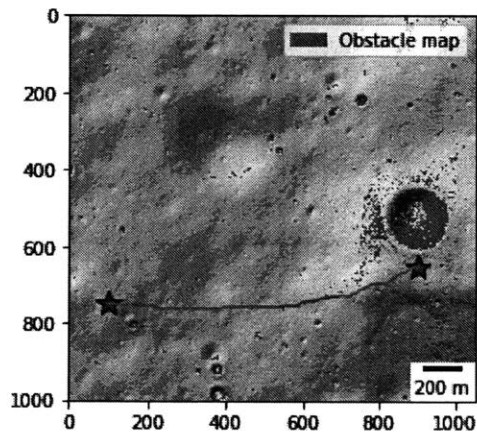
```
[8] pathfinder = astarSolver(apollo14_model, agent, inhouse=False)
```

```
[9] apollo14_model = load_legacy("Apollo14.txt").loadSubSection(maxSlope=15,
pathfinder = astarSolver(apollo14_model, agent, inhouse=False)
```

```
[10] %%time
out,_,_ = pathfinder.solvemultipoint(waypoints)
```

Wall time: 43.3 s

```
[11] hillshade(apollo14_model,1)
plt.imshow(apollo14_model.obstacle_mask(), alpha=0.5, cmap='bwr_r')
plt.plot(*waypoints.to(apollo14_model.COL_ROW), \
         linestyle='None', marker='*', markeredgcolor="k", markersize=15)
plt.plot(*out.coordinates().to(apollo14_model.COL_ROW))
plt.legend(handles=[red_patch])
plt.gca().add_artist(ScaleBar(apollo14_model.resolution, location=4))
plt.show()
```



3. Post processing of a path

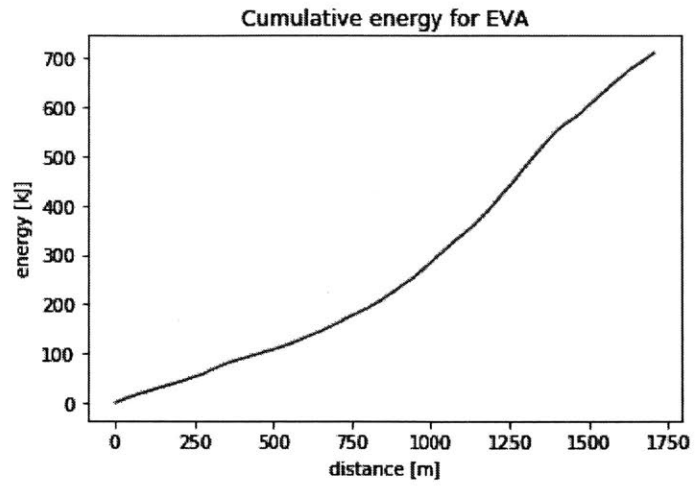
```
[12] from pexant.explorers import TraversePath
```

```
[13] traverse = TraversePath.frommap(out.coordinates(), apollo14_model)
```

```
[14] _, _, dr = agent.path_dl_slopes(traverse)
energies, v = agent.path_energy_expenditure(traverse)
```

```
[15] import numpy as np
```

```
[17] plt.plot(np.cumsum(dr), np.cumsum(energies)/1000.)
plt.xlabel('distance [m]')
plt.ylabel('energy [kJ]')
plt.title('Cumulative energy for EVA')
plt.show()
```



[]

Bibliography

- Abd Algfoor, Z., Sunar, M. S. and Kolivand, H. (2015), ‘A comprehensive study on pathfinding techniques for robotics and video games’, *International Journal of Computer Games Technology* **2015**.
- Aleksandrov, L., Maheshwari, A. and Sack, J. (2005), ‘Determining Approximate Shortest Paths on Weighted Polyhedral Surfaces’, *Journal of the ACM* **52**(1), 25–53.
- Anandapadmanaban, E., Tannady, J., Norheim, J., Newman, D. J., Hoffman, J. A. and Science, C. (2018), Holo-SEXTANT: an Augmented Reality Planetary EVA Navigation Interface, in ‘Proceedings of the 48th International Conference on Environmental Systems’.
- Bajracharya, M., Maimone, M. W. and Helmick, D. (2008), ‘Autonomy for Mars Rovers: Past, present, and future’, *Computer* **41**(12), 44–50.
- Bakker, N. H., Werkhoven, P. J. and Passenier, P. O. (1999), ‘The effects of proprioceptive and visual feedback on geographical orientation in virtual environments’, *Presence* **8**(1), 36–53.
URL: <http://www.mitpressjournals.org/doi/abs/10.1162/105474699566035>
- Carr, C. E., Newman, D. J. and Hodges, K. V. (2003), ‘Geologic traverse planning for planetary EVA’, *33rd International Conference on Environmental Systems (ICES)* (724), 2003–2416.
- Chappell, S. (2004), ‘Energetics and Resource Optimization for Planetary EVA Operations in Steep Terrain’, *Space 2004 Conference and Exhibit* (September), 1–11.
URL: <http://arc.aiaa.org/doi/10.2514/6.2004-6078>
- Collins, A., Adams, M. J. and Pew, R. W. (1978), ‘Effectiveness of an interactive map display in tutoring geography’, *Journal of Educational Psychology* **70**(1), 1–7.
- Couclelis, H., Golledge, R. G., Gale, N. and Tobler, W. (1987), ‘Exploring the anchor-point hypothesis of spatial cognition’, *Journal of Environmental Psychology* **7**(2), 99–122.
- Daniel, K., Nash, A., Koenig, S. and Felner, A. (2010), ‘Theta*: Any-angle path planning on grids’, *Journal of Artificial Intelligence Research* **39**, 533–579.

- Deans, M., Marquez, J., Cohen, T., Miller, M., Deliz, I., Hillenius, S., Hoffman, J., Lee, Y., Lees, D., Norheim, J. and Lim, D. (2017), Minerva: User-centered science operations software capability for future human exploration, in ‘IEEE Aerospace Conference Proceedings’.
- Deits, R. and Tedrake, R. (2015), ‘Footstep planning on uneven terrain with mixed-integer convex optimization’, *IEEE-RAS International Conference on Humanoid Robots 2015-Febru*, 279–286.
- Douglas, D. H. and Peucker, T. K. (1973), ‘Algorithms for the Reducation of Points Required to Represent a Digitized Line or its Caricature’, *Cartographica* **10**(2), 112.
URL: <https://search.ebscohost.com/login.aspx?direct=true&db=edb&AN=52215903&site=eds-live&scope=site>
- Essenburg, J. (2008), Mission planning and navigation support for lunar and planetary exploration, M.s. thesis, Massachusetts Institute of Technology.
URL: <http://dspace.mit.edu/handle/1721.1/46803>
- Ferguson, D. and Stentz, A. (2006), ‘Using interpolation to improve path planning the field D* algorithm’, *Journal of Field Robotics* **23**(2), 79–101.
- Galin, E., Peytavie, A., Maréchal, N. and Guérin, E. (2010), ‘Procedural generation of roads’, *Computer Graphics Forum* **29**(2), 429–438.
- Gilkey, A., Kobrick, R., Galvan, R., Johnson, A., Hoffman, J., Newman, D. and Melo, P. (2011), ‘Evaluation of a Surface Exploration Traverse Analysis and Navigation Tool’, *41st International Conference on Environmental Systems* (July), 1–9.
- Golledge, R. G. (1999), *Wayfinding behavior: cognitive mapping and other spatial processes*, Vol. 41.
- Johnson, A. W., Hoffman, J. A., Newman, D. J., Mazarico, E. M. and Zuber, M. T. (2010a), An Integrated Traverse Planner and Analysis Tool for Future Lunar Surface Exploration, Master thesis, Massachusetts Institute of Technology.
URL: <http://hdl.handle.net/1721.1/59560>
- Johnson, A. W., Hoffman, J. A., Newman, D. J., Mazarico, E. M. and Zuber, M. T. (2010b), An Integrated Traverse Planner and Analysis Tool for Planetary Exploration, Master thesis, Massachusetts Institute of Technology.
- Johnson, B. J. (2008), Optimization of a Planning Support System for Planetary Exploration Extravehicular Activities, Technical report, Massachusetts Institute of Technology.
- Karaman, S. and Frazzoli, E. (2011), ‘Sampling-based algorithms for optimal motion planning’, *International Journal of Robotics Research* **30**(7), 846–894.

- Karaman, S., Walter, M. R., Perez, A., Frazzoli, E. and Teller, S. (2011), ‘Anytime motion planning using the RRT’, *Proceedings - IEEE International Conference on Robotics and Automation* (May), 1478–1483.
- Koenig, S. and Likhachev, M. (2002), ‘D* Lite’, *Proceedings of the Eighteenth National Conference on Artificial Intelligence* pp. 476–483.
- LaValle, S. M. (1998), ‘Rapidly-Exploring Random Trees: A New Tool for Path Planning’, *In* **129**, 98–111.
URL: <http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Rapidly-exploring+random+trees:+A+new+tool+for+path+planning#0>
- Lee, S. Y., Lees, D., Cohen, T., Allan, M., Deans, M., Morse, T., Park, E. and Smith, T. (2013), ‘Reusable science tools for analog exploration missions: XGDS Web Tools, VERVE, and Gigapan Voyage’, *Acta Astronautica* **90**(2), 268–288.
- Lee, W. C. and Cheng, B. W. (2008), ‘Effects of using a portable navigation system and paper map in real driving’, *Accident Analysis and Prevention* **40**(1), 303–308.
- Likhachev, M., Ferguson, D., Gordon, G., Stentz, A. and Thrun, S. (2005), ‘Anytime Dynamic A*: An Anytime, Replanning Algorithm’, *Science* pp. 262–271.
- Lindqvist, L. V. J. (2008), Multidisciplinary Extravehicular Activity Mission Optimization for Lunar Exploration, M.s. thesis, Technische Universität München, Munich, Germany.
- Marquez, J. J. (2007), Human-Automation Collaboration: Decision Support for Lunar and Planetary Exploration, Phd, Massachusetts Institute of Technology.
- Mitchell, J. S. B. and Papadimitriou, C. H. (1991), ‘The weighted region problem: finding shortest paths through a weighted planar subdivision’, *Journal of the ACM* **38**(1), 18–73.
URL: <http://portal.acm.org/citation.cfm?doid=102782.102784>
- Muehlberger, W. R. (1981), Apollo 16 Traverse Planning and Field Procedures, *in* ‘Geology of the Apollo 16 Area, Central Lunar Highlands’, p. 11.
URL: <https://www.hq.nasa.gov/alsj/a16/traverse.pdf>
- NASA (2005), ‘NASA Exploration Systems Architecture Study’, (NASA-TM-2005-214062).
- NASA JSC (2009), *Dra 5.0*, number July.
- National Parks Service (1996), ‘Standards for trail construction’, *A Handbook for Trail Design, Construction, and Maintenance* pp. 29–33.
- NCALM (2007), ‘HAWAII Big Island Survey Report LIDAR System Description and Specifications’.

- Orz, F., Lofgren, G. E., Gruener, J. E., Eppler, D. B., Skinner, J. A., Fortezzo, C. M., Graf, J. S., Bluethmann, W. J., Seibert, M. A. and Bell, E. R. (2013), 'The traverse planning process for D-RATS 2010', *Acta Astronautica* **90**, 254–267.
- Oulasvirta, A. (2005), 'The Fragmentation of Attention in Mobile Interaction, and What To Do With It', *Interactions* .
- Passmore, R. and Durnin, J. (1955), 'Human energy expenditure', *Physiological Reviews* **35**(4), 801–840.
- RedBlobGames (2016), 'Introduction to A*'.
URL: <https://www.redblobgames.com/pathfinding/a-star/introduction.html>
- Santee, W. R., Allison, W. F., Blanchard, L. A. and Small, M. G. (2001), 'A proposed model for load carriage on sloped terrain', *Aviation Space and Environmental Medicine* **72**(6), 562–566.
- Santee, W. R., Small, M. G. and Blanchard, L. a. (2003), 'Application of Energy Cost Algorithms for Load Carriage to Field Data', *Journal of the Human-Environment System* **6**(2), 69–76.
- Shah, Brual C, S. K. G. (2016), 'Speeding Up A * Search on Visibility Graphs Defined over Quadrees to Enable Long Distance Path Planning for Unmanned Surface Vehicles', *Proceedings of the Twenty-Sixth International Conference on Automated Planning and Scheduling (Icaps)*, 527–535.
- Shean, D. E., Alexandrov, O., Moratto, Z. M., Smith, B. E., Joughin, I. R., Porter, C. and Morin, P. (2016), 'An automated, open-source pipeline for mass production of digital elevation models (DEMs) from very-high-resolution commercial stereo satellite imagery', *ISPRS Journal of Photogrammetry and Remote Sensing* **116**, 101–117.
- Speyerer, E. J., Lawrence, S. J., Stopar, J. D., Gläser, P., Robinson, M. S. and Jolliff, B. L. (2016), 'Optimized traverse planning for future polar prospectors based on lunar topography', *Icarus* **273**, 337–345.
- Srinivasan, R. and Jovanis, P. P. (1997), 'Effect of selected in-vehicle route guidance systems on driver reaction times.', *Human factors* **39**(2), 200–15.
URL: <http://www.ncbi.nlm.nih.gov/pubmed/9302888>
- Thorndyke, P. W. and Hayes-Roth, B. (1982), 'Differences in spatial knowledge acquired from maps and navigation', *Cognitive Psychology* **14**(4), 560–589.
- Urmson, C., Anhalt, J., Bagnell, D., Baker, C., Bittner, R., Dolan, J., Duggins, D., Ferguson, D., Galatali, T., Geyer, C., Gittleman, M., Harbaugh, S., Hebert, M., Howard, T., Kelly, A., Kohanbash, D., Likhachev, M., Miller, N., Peterson, K., Rajkumar, R., Rybski, P., Salesky, B., Scherer, S., Woo-seo, Y., Simmons, R., Singh, S., Snider, J., Stentz, A., Whittaker, W. R., Ziglar, J., Struble, J. and

Taylor, M. (2007), Tartan Racing: A Multi-Modal Approach to the DARPA Urban Challenge, Technical report.

Willis, K. S., Hölscher, C., Wilbertz, G. and Li, C. (2009), 'A comparison of spatial knowledge acquisition with maps and mobile maps', *Computers, Environment and Urban Systems* **33**(2), 100–110.

URL: <http://dx.doi.org/10.1016/j.compenvurbsys.2009.01.004>

Wood, B. M. and Wood, Z. J. (2006), 'Energetically optimal travel across terrain: Visualizations and a new metric of geographic distance with anthropological applications - art. no. 60600F', *Visualization and Data Analysis 2006* **6060**, F600–F600.