

Safe Data Gathering in Physical Spaces

Sankalp Arora
CMU-RI-TR-18-64

October 2018

Thesis Committee:
Sebastian Scherer, CMU RI (chair)
William (Red) L. Whittaker, CMU RI
David Wettergreen, CMU RI
Kostas Alexis, UNR

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy

Copyright © 2018 Sankalp Arora

PUBLISHED BY ROBOTICS INSTITUTE, CARNEGIE MELLON UNIVERSITY, 15213

Abstract

Reliable and efficient acquisition of data from physical spaces has widespread applications in industry, policy, defense, and humanitarian work. Unmanned Aerial Vehicles (UAVs) are an excellent choice for data gathering applications, due to their capability of gaining information at multiple scales. A robust data gathering system needs to reason about multi-resolution nature of information gathering while being safe, and cognizant of its sensory and battery limitations. The state of the art algorithms with provable worst-case guarantees are unable to present an efficient solution online. This thesis addresses three critical aspects of enabling safe, efficient, and multi-resolution data gathering: (1) online budgeted multi-resolution informative path planning (IPP), (2) guaranteeing safety and, (3) optimization of sensing bandwidth for implicit and explicit data gathering requirements.

First, we present an online navigation algorithm to guarantee the safety of the robot via an Emergency Maneuver Library (EML). We define a vehicle to be safe in static environments if it can stay in known unoccupied space while operating in partially known environments. Finding an optimal solution online for a non-holonomic system with non-linear dynamic constraints, in an online fashion is computationally infeasible. We present an efficient method to construct an EML that fully exploits the vehicle's dynamics capabilities and known unoccupied space available to ensure safety at high speeds. Another advantage of the EML is that it defines a pertinent volume from which uncertainty needs to be removed, to ensure UAV's safety. Further, we present a sensor motion planning approach that optimizes mission costs while using EML to ensure vehicle safety by gaining information relevant to the mission.

Second, we prove that for a specific class of information gathering problems, which consist of informative actions, such as gaining elevation to gather low-resolution data, traditional Markov Decision Process-based approximate solvers are not optimal. In such cases, the belief space dynamics need to be modeled to obtain efficient solutions to the multi-resolution IPP problems.

Third, we present Randomized Anytime Orienteering (RAOr), an anytime, asymptotically near-optimal algorithm, that enables solving aforementioned multi-resolution IPP problems online by taking heuristically guided random walks in the space of near-optimal routes. Although the work focusses on developing motion planning algorithms, we also describe representations that enable these planning algorithms to run on-board on computationally constraint platforms.

The algorithms developed, form a framework for safe, efficient, multi-resolution data gathering that has enabled UAVs to operate in diverse environments, scales, and applications. Further, we evaluate our algorithms on multiple UAVs varying from full-scale helicopters to small quad-rotors, running closed-loop autonomous missions that cumulatively span hundreds of kilometers.

Acknowledgements

My first thanks to my advisor, Sebastian Scherer. His enthusiasm for robotics systems, guidance, and support both technical and otherwise led to the generation of this body of work. I am thankful to William "Red" Whittaker for inspiring me to study robotics, and for his insights about the impact and methods of research. I am grateful to David Wettergreen for teaching me how to write a thesis and keeping me on-track towards finishing. I am indebted to Alexis Kostas for in-depth technical discussions on active data gathering and guiding me through the journey from proposal to defense. My thanks to Drew Bagnell and Mike Erdman for teaching me the fundamentals of robotics, attending your lectures was a joy.

I have had the tremendous good fortune to have built algorithms and robots with folks at AIRLab - Andrew, Hugh, Tomoyuki, Luke, Brian, Kristen, Srikanth, Jay, and Azarakhsh, thanks, I learned a lot working with you. My thanks to Sezal, Greg, Song, and Steve without whom MAVScout project would not have been possible. My thanks to Silvio for his constant encouragement and his calming presence that encouraged me to keep my head down and keep working. Thanks to Ratnesh, Sam and Rogerio for bringing the energy and excitement of youth to our research discussions. I am grateful to Daniel Maturana for his friendship, advice, in-depth research discussions and late-night rants, most of all thanks for bearing with my rants and code during MAVScout. Thanks to Daniel Althoff for teaching me to be rigorous in math, make pretty graphs, write papers, and being a wonderful mentor, I treasure your friendship. I am indebted to Sanjiban for his mentor-ship, guidance, life-advice and most of all, for his friendship, sharing this journey with you has been a privilege.

I owe many thanks to Sanjiv Singh, for bringing me to CMU and giving me the opportunity of working on dream projects. I am grateful to Joseph Djugash, Bradley Hamner, Debadeepta Dey, and Marcel Bergerman for their mentor-ship during my initial days at CMU. Lyle, your initial ideas about active perception were critical in developing this work. Thanks to AJ and Adam for their help in making AACUS a reality. My thanks to Sanae Minick, Nora Kazour, Suzzane Lyons-Muth and Allison Day, for fabulous administrative support.

Thanks to Teja Sukhavasi and Ali-Akbar Agha-Mohammadi for sharing the vision of active data gatherers and providing a platform to share our research at Qualcomm. Thanks to Robb Myer, Dave Mawhinney, Kit Needham, Allyson Hince and, Swartz center staff for showing faith in our team to develop active data gatherers that will solve real-world problems. Your confidence gave me a second wind. Thanks to Stuart Evans for his mentor-ship and guidance.

Thanks to my friends Zania, Qi, Rachel, Alok and, Aayush for your belief, support and for being a family away from home.

Thanks to Geetesh for being a friend, a colleague, a co-developer, a fellow researcher, and partner in crime since high-school. I have learned hard-work, humility, perseverance, and many life-lessons from you.

Words will not do justice in expressing my gratitude to my family. My parents, grandparents, uncle, aunt, and siblings their unconditional love, sacrifice and support are the pillars around which my life is built. This thesis is as much theirs as it is mine.

Finally, thanks to Prerana, my partner, my friend, my strength and my joy in this journey. You have been a constant

companion through the ups and downs of the last fourteen years, and I could not have asked for a better one. Thanks for guiding me to land and give me wings again whenever the air was too turbulent to handle.

This thesis was funded by Office of Naval Research, Qualcomm Innovation Fellowship, and Swartz Innovation Fellowship. Their support is gratefully acknowledged.

Contents

1	Introduction	21
1.1	Motivating Examples	22
1.2	Saliencies of Physical Data Gathering	23
1.3	Challenges	24
	State Of The Art	24
	Myopic Active Data Gathering - Heuristic driven Policies	24
	Non-Myopic Active Data Gathering - Offline Computed Policies	25
	Decision Theoretic Data Gathering - Optimal but Intractable	25
1.4	Thesis Statement	26
1.5	Contributions	26
	Ensuring Safety through a Library of Emergency Maneuvers	26
	Sensor Motion Planning for Implicit and Explicit Data Gathering	27
	Explicit Data Gathering with Path-Length Constraints	27
	Experimental Results	28
1.6	Document Outline	28
2	Background	31
2.1	Decision Theoretic Formulation of Safe Data Gathering	31
2.2	Related Work	32
	Active Data Gathering	32
	Ignoring Motion Constraints - Sensor Motion Planning	32
	Ignoring Motion Constraints - Vehicle Motion Planning	33
	Modelling Motion Constraints	34
	Guaranteeing Safety	34
	Data Gathering Systems	35
2.3	Developing Algorithms that Enable Safe Data Gatherers in Physical Spaces	36
3	Ensuring Safety in Partially Known Environments	37
3.1	Safety Definition	38
3.2	Approach	40
	Offline Generation of a Library of Emergency Maneuvers	40
	Modeling Safety as Survivability	40
	Monotonicity Proof	41
	Sub-Modularity Proof	42

	Greedy Algorithm for Generating Emergency Maneuvers	43
	Using Emergency Maneuvers to Ensure Safety	44
3.3	Results	45
	Reduction in Sensing Requirements	45
	Case Study 3.1: Safety Execution Test	46
	Case Study 3.2: Sensor Failure	47
	Case Study 3.3: Quantico and Mesa Flights	48
3.4	Summary	50
	Contributions towards Enabling Safe Data Gathering in Physical Spaces	51
4	Sensor Planning for Implicit and Explicit Data Gathering	53
4.1	De-Coupling Sensor and Vehicle Motion Planning	54
4.2	Approach	57
	Contextual Importance Function	58
	Pertinent Information for Implicit Data Gathering	58
	Pertinent Information for Explicit Data Gathering	58
	Policy Parametrization for Sensor Planning	59
	Input Features for Sensor Planning Policy	59
4.3	Policy Search	60
	Worst Case Policy Search Scenario for Implicit Data Gathering	61
	Policy for Explicit Data Gathering	65
4.4	Results	66
	Effective Range Increment	66
	Case Study 4.1: A Typical Landing Mission	67
	Case Study 4.2: Quantico and Mesa Flights	68
4.5	Summary	69
	Contributions towards Enabling Safe Data Gathering in Physical Spaces	70
4.6	Proofs and Algorithmic Details	72
	Maximizing Information Gain to Minimize Trajectory Cost	72
	Complexity of Calculating Information Gain	72
	Calculating Features	73
	Worst Case Scenario for Safety	75
	Monotonicity of Expected Information Gain	77
	Probability of Detection vs. Distance	77
	Reduction in Computational Complexity for Evaluating Actions	78
5	Modeling of Belief Space Dynamics for Budgeted, Multi-Resolution Data Gathering	81
5.1	Explicit Data Gathering	83
	MDP Formulation	83
	POMDP Formulation	84
5.2	Solving POMDP through MDP Solvers	85
5.3	Expected Value of Information	86
5.4	Examples	88
	Tiger Problem	88

Multi-resolution, Budgeted Information Gathering	89
5.5 Summary	91
Contributions towards Enabling Safe Data Gathering in Physical Spaces	91
6 Randomized Algorithm for Path Planning for Budgeted Data Gathering	93
6.1 Information Theoretic Formulation for Explicit Data Gathering	94
6.2 From Data Gathering to Orienteering to Set Selection and TSP	95
Constraint Satisfaction Problem	96
Traveling Salesman Problem	96
6.3 Randomized Anytime Orienteering	96
Drawbacks	98
Uniform Sampling of Sets	98
Sampling Inadmissible Sets	98
6.4 Randomized Anytime Orienteering - Greedy (RAOr-G)	98
Local Greedy Search Heuristic	99
Weighted Sampling	100
6.5 Results	101
Computational Performance	101
Correlated Rewards	102
GCB vs RAOr-G	103
Case Study 6.1: Exploration mission with low budget	103
Case Study 6.2: Flight tests at Gascola	104
6.6 Summary	104
Contributions towards Enabling Safe Data Gathering in Physical Spaces	106
6.7 Proofs	107
6.8 Representations & Systems	110
Micro-Aerial Vehicle Platforms	110
Semantic Grid Representation	111
Local Perception & Planning Pipeline	115
7 Conclusion	127
7.1 Results Summary	127
7.2 Significance of Contributions	128
7.3 Discussion	130
Survivability Model for Safety	130
Trajectory Cost Minimization & Information Gain Maximization for Sensor Motion Planning	131
Presence of Informative Actions in Data Gathering	132
Independence of Reward Function For Explicit Data Gathering from Order of Viewpoints	132
7.4 Future Work	132
Combining Implicit and Explicit Data Gathering	132
Safe Local Informative Path Planning	134
Multi-Heuristic Global Path Planning	139
Selecting the Set of Heuristics	140
Data Gathering for Improving Long-Term Deployment	141

7.5 Final Thoughts	143
Bibliography	145

List of Figures

- 1.1 Motivation for multi-resolution information gathering. 22
- 1.2 Motivation for semantically guided exploration. 23

- 3.1 Safety Algorithm: (a) A motion planner sends a planned trajectory to the safety algorithm (b) The safety checker queries an emergency maneuver library to ensure maneuvers exist from the trajectory that lie entirely in known free space. (c) Eventually the safety checker encounters a case where the library is unable to find a maneuver for the trajectory (d) The safety checker tries to slow down the vehicle until the library is successful. If not, the safety checker reverts to the previous guaranteed safe trajectory. 44
- 3.2 Generation of emergency maneuver library for one state. From left to right the plots step through the generation of emergency maneuver library for 6 iterations. The top row displays the search space from which the current trajectory is picked, where each trajectory is colored according to the probability of not passing through an obstacle in the set. The middle row shows the greedily selected maneuver in the current step in green and existing maneuvers in the set in black. The bottom row shows the total probability of finding at least one maneuver in the set not passing through an obstacle. The robot starts at 25m/s longitudinal velocity for all the maneuvers and for illustration purposes, is restricted to move in the xy plane. The benefit of adding new maneuvers diminishes as more trajectories are added and almost levels out after 5 trajectories. 46
- 3.3 Changes in Sensor Requirements. The sensor range required for safe operation of the vehicle when using stopping distance for safety is displayed in red, in green is the sensor range required for safe operation of the vehicle when using emergency maneuver library for helicopter safety. 47
- 3.4 Data from a flight test conducted on 18th December 2013 in Manassas, Virginia.
 - a) Helicopter approaches a large simulated wall with the emergency trajectory libraries with no emergency maneuver in contact with the wall. b) As the helicopter gets closer to the wall, the emergency maneuvers intersect the wall and become invalid. Only valid maneuvers are displayed. c) More emergency maneuvers are pruned away as they come in contact with the wall d) An emergency maneuver is executed as the future state is no longer safe. 47

- 3.5 Guaranteeing safety even when the sensor fails (a) A situation where the sensor has failed. In such a situation, the space that is unknown will remain unchanged due to lack of updates from the sensor. However, the current trajectory being followed is guaranteed safe by construction as it always lies in the space that is known to be free (b) The robot eventually executes the evasive maneuver as the unknown space remains unchanged. This ensures the robot perpetually remains in a safe state. 47
- 3.6 Safety Quantization: Flight test in Quantico, Virginia. a) Shows an autonomous landing mission conducted in Quantico, Virginia on Unmanned Little Bird. b) Shows the safe velocity of the helicopter with the emergency maneuver library during the flight tests in dashed line, the executed velocity in solid line and the safe velocity if stopping distance is used in dotted line. c) This figure shows planning time available to the planner, before the vehicle will reach the edge of known space and execute one of the emergency maneuvers. The planning time calculated assuming the helicopter will follow the current planned trajectory. 49
- 3.7 Different planning problem distributions in different environments. The lines denote the path traced by the helicopter superimposed on a satellite image (a) Missions in Quantico, VA, where obstacles are towers, trees and no fly zones (b) Missions in Mesa, AZ, where obstacles are mountains and no fly zones. 49
- 3.8 The figures show the allowed plan time, distance to the landing zone (LZ) and, the safe velocity relative to the executed velocity of the helicopter. The black line shows the mean and the gray dashed line illustrates the upper and the lower bound of the measurements of all considered flight tests. 50
- 4.1 Application Scenario: Top Left – Test Vehicle, Boeing Unmanned Littlebird. Top Right – Near Earth Autonomy M3 sensor suite, equipped with a actively controllable high range laser. Bottom – Example mission scenario, the vehicle is suppose to autonomously navigate to a pre-defined landing zone at high speeds, evaluate it and decide to whether to land or not while ensuring safety. 55
- 4.2 Approach Overview: The sensor is actively controlled through a policy that takes in the features that are extracted from robot’s belief. The features take expected information gain and contextual importance function into account. 57
- 4.3 Contextual Importance Function – The figure illustrates the region with contextual importance function greater than 1 in orange, and region which is known to the robot in grey. As the robot navigates, the known region has no more information to be gained that might affect its future actions. The volume around the trajectory, bounded by the emergency maneuver library for the future unsafe states and volume inside the landing zone is contextually important. It is for gaining this information that we need to optimize the sensory actions. 59

- 4.4 Information Gain – The expected information gain given a prior ($p(o)$) is monotonic in probability of detecting the obstacle if the obstacle is present, $p(d|o)$. Assuming there are no false positives, $p(d|o') = 0$. This implies for maximizing information gain, one may maximize $p(d|o)$ 62
- 4.5 Expected Probability of Detection: The expected probability of detection of a wire if it exists, $\mathbb{E}_{p(o)} p(d|o)$ for a given action is an indicator of how good an action is. The plots show $\mathbb{E}_{p(o)} p(d|o)$ versus varying nodding time period for the worst case scenario for the sensor at vehicle velocity, $v = 45 \text{ m/s}$, for a uniform $p(o)$. Each sensor velocity corresponding to different nodding time periods is evaluated till $t_r = 1.4\text{s}$. The evaluation shows scans with slower scanning speed are better, this is intuitively the correct behavior as slower nodding speeds means more uniform point distribution of rays in the $[\theta, \phi]$ manifold. Similarly a lower fast axis resolution (Yaw Res.) results in a more uniform distribution of rays in the $[\theta, \phi]$ manifold, leading to more information gain. 65
- 4.6 Left: Effective sensor range increment using the worst-case policy as compared to passive scanning. Right: Effective sensor ranges and max safe speeds for online (PASP), worst-case and passive sensing policies. 66
- 4.7 A mission at Quantico, 26th Feb 2014 68
- 4.8 Sensor Nodding Profile: The figure shows the sensor's nodding profile. In the initial part of the mission, the sensor acts to keep the vehicle safe. It switches to focussing on the landing zone when safety for the remainder of the mission has been guaranteed. This focussing of the laser is shown by the narrow peak to peak of the nodding angles. At the very end, once enough points on the landing zone have been focussed, the sensor reverts back to ensuring that the vehicle can be safe should it desire to waveoff. 68
- 4.9 Safety Benefits: The figure shows the sensor's nodding profile. In the initial part of the mission, the sensor acts to keep the vehicle safe. It switches to focussing on the landing zone when safety for the remainder of the mission has been guaranteed. This focussing of the laser is shown by the narrow peak to peak of the nodding angles. At the very end, once enough points on the landing zone have been focussed, the sensor reverts back to ensuring that the vehicle can be safe should it desire to waveoff. 68
- 4.10 Mission Definition: The helicopter navigates from loiter point (on the right) to landing zone (on the left), a distance of more than 10km in less than 210 seconds. It has to navigate the environment while being provably safe and touch down at the LZ without hovering over it to look for potential sites. The sensor is controlled to enable safe completion of the mission. 69
- 4.11 Mission Definition: The rotorcraft has to navigate the environment while being provably safe and touch down at the LZ without hovering over it to look for potential sites. The sensor is controlled to enable safe completion of the mission. 69

- 5.1 (a) The UAV gets a high reward if it visits the same cell as the car to gather high resolution information of the car (lets say its numberplate.). Since the UAV knows where the car is, it does not need to take action up and can directly go to the cell with the car. (b) In the POMDP version of the problem the location of the car is unknown, hence there is only a small chance that the next cell the UAV goes to has a car in it, leading to a small expected reward. (c) But if UAV gains height, the uncertainty about the location of the car is removed. Although there is no reward from this action but the removal uncertainty leads to guarantee of gaining high reward by visiting the cell in which the car exists. Since, MDP-POMDP solvers cannot take informative action up , they are sub-optimal for such data gathering problems. 89
- 6.1 An illustration how RAO-G combines CSP and TSP solvers in combination with local greedy heuristics to explore the space of routes rapidly, resulting in improvement of run times for finding near optimal solutions for the correlated orienteering problem. a) Using the CSP algorithm the current admissible route is given by $r = x_{start}, x_1, x_2, x_3, x_{end}$. b) At the next step x_4 is sampled, and is to be added to the route. c) The exponential number of ways in which x_4 can be added to r is reduced to a near optimal order in polynomial time by the TSP solver. This step reduces an exponential search space with in polynomial computation costs. d) The new route obtained is then improved by conducting a local search using greedy heuristics. 97
- 6.2 Illustration of drawbacks of RAO algorithm. a) The problem shown in here consists of three high valued nodes and a budget just sufficient to visit all three. The optimal solution is to visit all three nodes but very few routes exist that have a reward close to optimal. The solution shown is found by running RAO-G algorithm, Alg. 9. b) Shows the run time vs reward plot for the problem shown in (a) for both RAO-G and RAO. RAO can only attain approximately 66% of the optimal value, as it has a hard time selecting the correct set of nodes. c) Illustrates the problem RAO faces while running on problems with a small budget. Here a 5X5 grid of nodes, distributed uniformly at a resolution of 1 with a budget of 15 is shown. The routes in red are routes sampled by RAO that exceed the budget, whereas routes in blue are routes that were sampled that do not exceed the budget. Visibly, red routes outnumber the blue routes. d) The same problem leads to poor run times of RAO for relatively low budget problems. Here we show the run times of RAO and RAO-G on the 5X5 grid shown in (c) while varying the available budget. A runtime of 0 signifies that the algorithm did not converge. Clearly RAO did not converge for low budget problems in the allotted time. Highlighting the limitation of RAO in dealing with low budget problems. 98

- 6.3 Illustrated here is 10X10 grid size benchmark problem. For all the benchmark problems the nodes are situated in a uniform grid with 1 resolution. For this particular problem *RAOr-G* is able to find a near-optimal solution in 6.9 seconds while the state of the art takes 143.6 seconds for finding same quality of solution. 101
- 6.4 a) *RAOr-G* planned path with a 500m budget on the 100X100 area for *Case2*. Red marks the sensor footprint. Notice how the path visits high value regions, displayed in black. Grey paths show all the paths searched by *RAOr-G*. b) and c) Case 1 is displayed in dashed line, Case 2 in solid lines. *RAOr-G* is competitive with greedy algorithms in Case 1 and dramatically outperforms greedy and RIG for Case 2, where greedy algorithms are stuck in local minima. 102
- 6.5 Performance comparison of GCB and *RAOr-G* for a scenario where gaining height is expensive and hence greedy algorithms tend to get stuck in local minima. 103
- 6.6 a) Testing site, start and end are marked by green nodes and car locations are shown in orange. b) Vehicle starts with a budget of 700m, the reward increases as likelihood of finding the car increases, the crest in reward marks the time at which the global planner found and decided to map the car. Figures 1, 2, 3 and, 4 show the series of plans at various stages of the exploration mission, Dark squares indicate absence of cars and red squares presence of cars. Shades of grey and red signify certainty. Once the car is recognized, a 360 view of the car is associated with a high reward. 104
- 6.7 Long-range exploration mission, where the robot was tasked to explore 150,000m² of area, and it completed the mission in 1 battery pack, just under 5 minutes. The top left image shows the scenario, and other 3 images show mission progress in chronological order as denoted by the flight time displayed in them. 105
- 6.8 All the locations where the robot has been successfully able to find and map cars in high resolution (Red dots). Start and end points of missions from which the robot began and ended scouting. (White Dots) 105
- 6.9 MAV System Block Diagram: Exploration Planner *RAOr* provides global trajectories to the disparity planner described in section 6.8. Semantic classification outputs classified images to the exploration planning node, that processes these images to construct a semantic grid map also described in section 6.8 111

- 6.10 Figures 1 shows the updated map after a classified image 4 is integrated in our current mapping pipeline, Figure 2 shows the updated map if the classified image is projected on the DEM without exploiting semantic knowledge and Figure 3 shows the updated map if the ray interdependence is not modelled. Dark squares indicate absence of cars and red squares presence of cars. Shades of grey and red signify certainty. Modelling ray interdependence and exploiting semantic knowledge leads to better modelling of uncertainties due to occlusions while providing an improved cell occupancy estimate. Figure 4 provides the sensitivity analysis of mapping performance vs. DEM height errors. 115
- 6.11 Planning pipeline based on inverse depth obstacle perception. The frontal expansion and back expansion are shown in pink and red point cloud around the original point cloud of pole. Planned path around the pole is also shown with the current robot position circled in green. 115
- 6.12 Left: Disparity vs Depth (blue) and probability distributions are shown in red and green. Red and Green PDF in disparity are same and easy to model but their corresponding Red and Green PDF in range vary and difficult to model. Hence we use inverse depth space to represent obstacles. Also, disparity i.e. inverse range captures space at multi-resolution suitable for registration of stereo sensor data. Right: Shows the pixel-wise expansion of a point obstacle according to robot size. 117
- 6.13 Disparity expansion shown as point cloud. The pink and red point cloud represent the foreground and background disparity limits. 118
- 6.14 Pose Graph of expanded disparity images. Dashed path shows robot motion and stored nodes in the graph are shown as triangles. Nodes are stored at intervals of distance and orientation. 119
- 6.15 Location where experiments were carried out: highlighted area 123
- 6.16 Time profile of expansion step. 124
- 6.17 Point cloud is shown at the bottom in all three figures for reference. Point cloud is colored by height in (a) & (b) and by actual intensity in (c). (a) Planned path(green) between low trees highlighted in red ellipses (b) Replanned(green path) as more observations are made, marked in red ellipse, (c) Long range planning horizon. The point cloud shows the noisy measurement but even noisy information allows to infer occupancy at long distances. 124
- 6.18 Reactive Planning at $4m/s$: Top image shows the robot has planned to go right with unseen obstacle marked in red ellipse. Bottom image: after banking right an obstacle obstructs the previous plan and a new plan avoiding it is generated. 125
- 7.1 Current vs proposed data gathering pipeline 133

- 7.2 **Left:** Motion planner operates independently of the safety library. Resulting in maximizing the distance from walls of the corridor and slower navigation speeds. **Right:** If the motion planner is cognizant of *EML* and safety constraints, it can identify a faster solution to fly through the corridor while being close to a wall. 133
- 7.3 **Left:** Vehicle flies a constant speed around the target to gather its data, while myopically looking forward as and when required. **Right:** The same approach does not work in this scenario, as there is a sharp turn in the path, leading to the vehicle missing essential information about the target as it ensures safety. 133
- 7.4 **1:** In this scenario, the viewpoints are $10m$ away from the target, the vehicle is equipped with a sensor with a range of $22m$ and field of view of 60 degrees, if a viewpoint views a unique sector of the cylinder, $+1$ reward is gained. The maximum speed of the vehicle is restricted to 3 m/s . **2:** An informative route planned by using *RAOr*. **3:** A safe informative route planned by using *SafeRAOr* using path independence assumption. **4:** A safe informative route planned by using *SafeRAOr* while modeling path dependence. Given a maximum heading rate of the vehicle, $\dot{\psi} = 1.06\text{ rad/s}$, the trajectory generated by assuming paths are independent leads to total time taken by $38.24s$, while the trajectory generated by modeling path dependence takes $34.37s$ to cover the same viewpoints. It takes the planner $1.12s$ to generate the trajectory. 136
- 7.5 Scouting of Landing Zones (LZ). We want the vehicle to be able to scout for landing zones. **Left:** Since we cannot deal with big problems online, we reduce the size of those problems by aggregating many nodes into one. In this case vehicle does not scan the LZ, because the cost of visiting the node is out of budget. **Right:** Faster global planning will enable increased granularity (more nodes) and hence enabling partial coverage of LZ. 139
- 7.6 Illustrated here is a scenario where wrong assumptions about the environment lead to an uncertain map, which requires the vehicle to travel more to gain information about the environment. The vehicle gains the same information about the environment with a much smaller path if its assumptions about the object heights in the environment are correct. 141

1

Introduction

Data acquisition is relatively easy in the virtual world, where the cost of accessing data can be equivalent to accessing a memory block. However, the cost of data gathering in physical spaces, where it is impractical to have sensor networks is not as trivial. Currently, we rely on humans to carry or drive sensors around to digitize the physical world to collect data.

Gathering data from an oil pipeline to predict future leaks, collecting data from a lake to sample spatial distribution of water contaminants to predict outbreak of diseases, gathering data about flood survivors to aid in search and rescue missions, or to collect data about bridges to predict corrosion rates, these applications require data gathering in the physical realm at an unprecedented scale. Using humans to do these tedious, boring and often risky tasks is far from ideal. Therefore, there is an urgent need to develop autonomous robots to do the job for us.

Micro-Aerial Vehicles (MAVs) are well suited for such data gathering tasks. The primary reason is their ability to reach vantage points unattainable by other systems. MAVs can switch from viewing large spaces at a distance to flying in close to obtain more accurate information. This capability of gaining information at different scales makes MAVs excellent for the applications mentioned above. Hence, the last decade has seen massive growth in research for applications for MAVs such as search and rescue [Goodrich et al., 2008], infrastructure inspection [Bircher et al., 2015], surveillance, crop and wildlife monitoring [Hodgson et al., 2016].

A common trend in these applications is that not all possible sensing locations are of equal value; we are usually more interested in gathering information about specific targets, such as victims, vehicles, animals. Often we do not know in advance the location of these targets, making it necessary to locate them before conducting more detailed inspection. We argue that we need to develop safe data gathering robots that can exploit their multi-resolution data gathering capabilities while reasoning about their motion and sensory constraints.

Section 1.1 motivates the need for such robots through an example and a field experiment. Section 1.2 describes the salencies of the problem

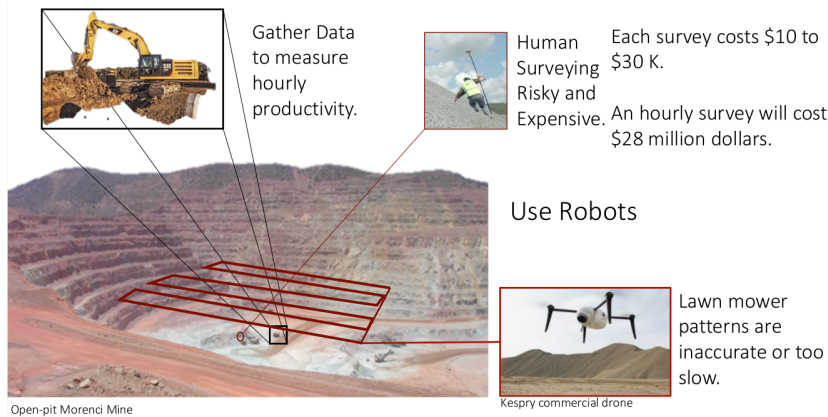


Figure 1.1: Typical scenario, where earth moving data is required on an hourly basis at a mining site to measure machine productivity. Currently, vehicles take a brute force approach to survey the complete site, leading to low-resolution data collection. For an MAV to collect relevant data on this site, it needs to find objects of interest and focus its sensory and physical resources in gaining data around those objects of interest. We will enable such data gathering through the techniques developed during our thesis.

of gathering data in physical spaces. The challenges of addressing these saliencies with respect to state of the art are described in section 1.3. Our thesis is presented in section 1.4, followed by a summary of contributions, section 1.5 and the document structure in section 1.6.

1.1 Motivating Examples

Mining companies are interested in measuring earth moved by earth movers operating on their sites on an hourly basis. Earth movers are only capable of altering the earth around them in that time. Nonetheless the current commercial data gathering MAVs use lawn mower patterns at a fixed height above the environment to gather data, as shown in figure 1.1. Often, the data collected from a height does not have enough resolution to compute the small amount of earth moved by the machinery. A better approach would be to focus the sensory bandwidth of an MAV on gathering low-resolution imagery to locate machines of interest and then capture high-resolution data around those machines of interest selectively.

We tested our hypothesis at Fort Indian Town Gap, figure 1.2, where we wish to collect high-resolution data of cars to make their 3D models. A lawnmower aerial surveying pattern of the whole sites takes 70 minutes to run, requiring multiple battery changes. The result of the run is a good looking 3D model of the environment, but there is not sufficient data to make a high-resolution model of the cars in the environment. Whereas, a multi-scale data gathering approach where the MAV gains height to gain information about the location of cars and then gains high-resolution data by focusing its sensing bandwidth on the cars leads to a high resolution 3D model of cars within 10 minutes of flight time.

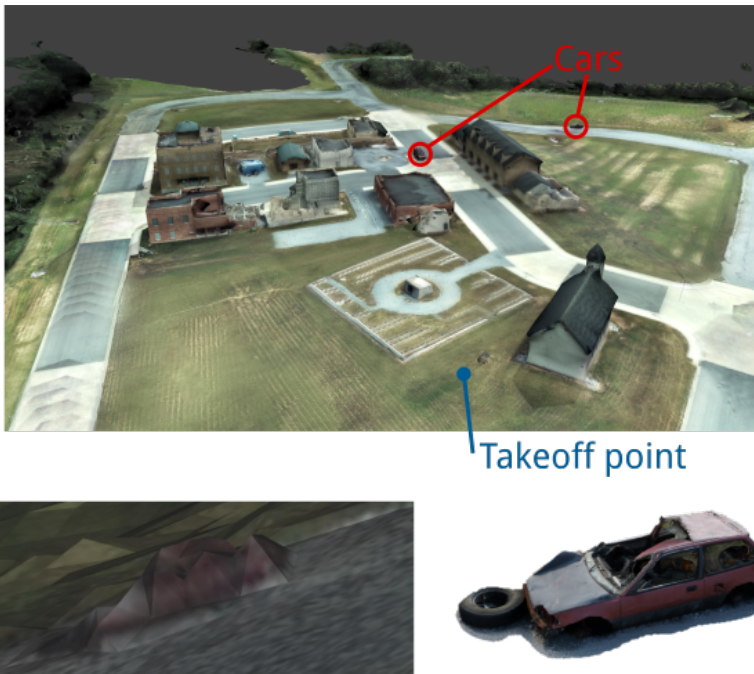


Figure 1.2: Top: Our test site, with the location of a car (our object of interest) and the MAV takeoff location highlighted. **Bottom left:** 3D Model of the car constructed with 1 hour and 10 minutes of “lawnmower” pattern flight. **Bottom right:** 3D Model of the car constructed with 10 minutes of human-piloted, focused flight, starting from the same location. The goal of safe, multi-resolution data gathering is to efficiently gather high-quality information pertinent to the user.

1.2 Saliencies of Physical Data Gathering

Above example shows that focussing sensory and physical resources of an agent while gathering data pertinent to the mission makes them an effective tool for data gathering. Mobile platforms pose constraints on the sensors, compute they can carry. Given the limited battery life, their operating time is also limited. Such constraints lead to the following four salient characteristics to the problem of safe, efficient data gathering in partially known environments.

1. *Safety Constraint.* The vehicle has to ensure that it will never run into obstacles while gathering data.
2. *Constraint on sensing bandwidth.* A limited payload carrying capacity limits the number of sensors the vehicle can carry leading to limited sensing bandwidth.
3. *Correlated nature of information.* Equipped with cameras, some mobile platforms like UAVs can view large areas from a distance to gain information. This capability leads to the reward of visiting different locations being correlated.
4. *Constraint on the total travel distance.* Due to limitations of fuel/battery,

the length of the route taken by the mobile platforms is limited.

Safety constraint necessitates that the agent gathers data about the environment to make sure its obstacle free, we call such a data gathering requirement as *implicit* and the act of gathering data to satisfy an implicit requirement as implicit data gathering.

Similarly, the need for task-specific data gathering is called *explicit*. For example, for a pipe inspection robot, the user explicitly specifies to gather data about cracks and rust on pipes.

Actively gathering implicit and explicit data while modeling physical constraints requires agents to reason about the high-dimensional uncertainty of the environment. We identify unique challenges posed by the problem in the next section.

1.3 Challenges

Active data gathering refers to optimization of data being gathered during experimentation. The data gathered is optimized to meet the objectives of the experiments. The desire is to meet experiment's objectives at minimum cost or to maximize a reward function. The search space for active data gathering is the space of all possible actions and possible data gathered from these actions. Since the data gathered from the actions are stochastic, the space of possible actions and possible observations can be large and active data gathering problem is NP-Hard in this large search space.

State Of The Art

In digital spaces, however, where the cost of gathering data is independent of data points gathered, the problem is well studied and even though its NP-Hard, near-optimal polynomial time solvers exist.

Unlike active data gathering in physical spaces, active data gathering in digital domain does not have to contend with motion constraints. Reasoning about motion constraints exponentially increase the search space. Majority of the current state of the art physical data gathering systems attempt to solve the physical data gathering problem using following three lines of approaches.

Myopic Active Data Gathering - Heuristic driven Policies

One approach is to ignore the motion constraints and apply a myopic strategy where a set of sensing locations is identified, and the agent travels to the most promising one. While such strategies are computationally efficient, they are generally heuristic driven and myopically optimize Pareto-optimal reward functions. On account of being myopic, these strategies inadequately model motion constraints like constraint on total path-length, leading to sub-optimal behavior.

Non-Myopic Active Data Gathering - Offline Computed Policies

Another approach is to solve for non-myopic policies that model motion constraints. Although such methods provide near-optimality guarantees, their runtimes do not enable online generation of solutions. These approaches ignore the need for gathering data for safety of the vehicle or make simplifying assumptions on vehicle dynamics, leading to the vehicle operating at slow speeds.

Decision Theoretic Data Gathering - Optimal but Intractable

Decision-theoretic approaches, use POMDP solvers to address the data gathering problem and have been able to address the active data gathering problem for localization successfully but the dimensionality of active data gathering for safety or mapping makes them unsuitable to run online.

We capture some of the challenges involved in developing safe, efficient, multi-resolution data gathering agents in the following list.

- *Challenge 1: Guaranteeing Safety while Fully Exploiting Vehicle Dynamics and Known Space* - The safety constraint requires the vehicle to be in a control invariant state that entirely lies in known free space. State of the art makes simplifying assumptions on the dynamics of the vehicle to meet these constraints online. An extreme but well-accepted approach is using straight line stopping to ensure safety. Such approaches lead to under-utilization of vehicle dynamics and sensory capabilities. Modelling non-linear dynamics and complex known space geometry online, in a fixed computation time, to enable the vehicle stay in known free space poses a challenge.
- *Challenge 2: Designing Online Algorithms for Active Data Gathering for the Implicit Need to Ensure Vehicle Safety* - The data gathering agent has to reason about gathering data about possible obstacles in the environment to guarantee its safety while also gathering information that is explicitly required by the user. Given the limited sensing bandwidth, the agent has to infer the pertinent information to be gathered for safety and plan sensory actions accordingly. Inferring pertinent information is dependent on the reachability of the vehicle, which is computationally intensive to compute for vehicles with non-linear dynamics. Moreover, computing expected information gain is computationally infeasible on mobile platforms. Making it non-trivial to design algorithms that enable active data gathering for the implicit need of ensuring vehicle safety.
- *Challenge 3: Designing Near-Optimal Algorithms for Explicit Data Gathering while Reasoning about Budget Constraints* - Reasoning about budget constraint while active data gathering requires the agent to reason about its future actions non-myopically. However, decision-theoretic data gathering requires planning in high dimensional belief spaces. Approximate

solutions to the decision-theoretic problem have been suggested, and have been able to reduce the run-time significantly, but their current computation requirements render those solutions to be infeasible online. Myopic or receding-horizon approaches have been successful at providing reasonable solutions to informative path planning, but they lead to oscillations, which negatively affects the performance of the vehicle under the budget constraint. Information theoretic path planning problem is also NP-Hard in a high dimensional information space. The high dimensional state (belief) space combined with the need for a non-myopic solution make the design of a near-optimal algorithm for explicit data gathering while reasoning about budget constraints challenging.

The work presented in this document focuses on addressing these challenges to develop efficient data gathering agents in physical environments.

1.4 Thesis Statement

The central thesis of this document is that - For mobile robots operating in partially known environments, active gathering data and concurrently reasoning about motion constraints produces higher rates of information gain and higher speeds of safe operation.

1.5 Contributions

The contributions of this thesis are three algorithms that can be used to address the safe data gathering problem in physical spaces while reasoning about motion constraints. The algorithms assume the data gathering agent is operating in a static, partially known environment and it can carry limited computational power. The three algorithms are designed to solve three different aspects of the safe, efficient data gathering namely enabling safe operations of mobile agents in partially known environments while utilizing their dynamic limits, reasoning about pertinent data gathering for the safety of the mobile agents while modeling their dynamic constraints and reasoning about task-specific, multi-resolution data gathering with budget constraints.

Ensuring Safety through a Library of Emergency Maneuvers

The first algorithm demonstrates a way to guarantee the safety of an agent operating in partially known static environments while utilizing its dynamic capabilities. The algorithm efficiently constructs a near-optimal library of emergency maneuvers by using survivability as a metric. Since this library is constructed offline, full non-linear dynamics of the vehicle can be exploited for the generation of the library. Moreover, the sub-modular and monotonic structure of survivability enables polynomial time solution of NP-hard library generation problem.

This library of emergency maneuvers is then used to enforce the safety constraint online. The resulting algorithm is agnostic of the rest of the planning pipeline and has a guaranteed upper bound on runtime. We show that the algorithm enables rotorcraft to fly safely at 1.8 times the speed as compared to techniques that approximate dynamics of the vehicle to ensure safety. [Arora et al., 2015, 2014]

Sensor Motion Planning for Implicit and Explicit Data Gathering

In the second algorithm, we describe a way to optimize sensor motion to gather, implicit information for ensuring vehicle’s safety, and explicit task-specific information, given the vehicle’s path is fixed. We show that decision-theoretic optimization of sensor trajectory for cost minimization of vehicle trajectory can be equivalent to information-theoretic optimization of the sensor trajectory for maximizing pertinent information gain. We use emergency maneuver library to infer the information pertinent for ensuring vehicle’s safety and describe a paradigm that optimizes sensory actions for worse case scenario for the safety of the vehicle. We show that these offline optimized sensory actions can be augmented online to improve vehicle performance further.

The developed algorithm provides performance bounds on the operating speeds of the vehicle. Computing the pertinent information for safety using emergency maneuvers enables the algorithm to run online while reasoning about the dynamics of the vehicle. We show that the algorithm enables 300% higher safe speeds for full-scale helicopters equipped with a long-range lidar than sensor motion planning schemes that fail to account for rotorcraft’s dynamic constraints. The algorithm assumes that the vehicle’s path is fixed; hence its scope is limited to mobile platforms for which there exists atleast a degree of freedom of sensor’s motion that can be controlled independently of vehicle’s motion. [Arora and Scherer, 2015]

Explicit Data Gathering with Path-Length Constraints

In the third algorithm, we focus on vehicle motion planning for a multi-resolution data gathering task while modeling path length constraints. We show that for budgeted, multi-resolution data gathering problems near-optimal solvers need to reason about belief space dynamics. Guided by this result, we describe a belief dependent reward function for the budgeted data gathering problem and present an anytime, asymptotically near-optimal budgeted data gathering solver (*RAOr*), that can efficiently solve for routes that maximize correlated reward functions subject to constraints on route cost. The key insight is to break the problem into the selection of viewpoints to visit and the order in which to visit them, leading to an exponential reduction in dimensionality. *RAOr* combined with expected observation assumption has enabled us to perform non-myopic, multi-resolution information gathering

on-board. [Arora and Scherer, 2017, Arora et al., 2018]

Experimental Results

The algorithms developed, form a framework for safe, efficient, multi-resolution data gathering that has enabled UAVs to operate in diverse environments and scales. We evaluate our algorithms on multiple UAVs operating at two scales, 1. Full-scale helicopters running closed-loop autonomous navigation and landing missions, 2. MAVs with gathering high-resolution data of objects of interest operating in unknown environments. The missions cumulatively span hundreds of kilometers of closed loop flights. Although the work has focussed on developing motion planning algorithms for information gathering, we briefly describe representations and systems that enable these planning algorithms to run onboard on computationally constraint platforms. [Choudhury et al., 2014, Dubey et al., 2017, Maturana et al., 2017]

1.6 Document Outline

Challenges	Chapters
<i>Challenge 1: Guaranteeing Safety while Fully Exploiting Vehicle Dynamics and Known Space</i>	<i>Chapter 3: Ensuring Safety in Partially Known Environments</i>
<i>Challenge 2: Designing Online Algorithms for Implicit Data Gathering to Ensure Vehicle Safety</i>	<i>Chapter 4: Sensor Planning for Implicit and Explicit Data Gathering</i>
<i>Challenge 3: Designing Near-Optimal Algorithms for Explicit Data Gathering while Modelling Motion Constraints</i>	<i>Chapter 5: Modeling of Belief Space Dynamics for Budgeted, Multi-Resolution Data Gathering</i> <i>Chapter 6: Randomized Algorithm for Path Planning for Budgeted Data Gathering</i>

Table 1.1 provides a mapping from chapters of this document to the challenges faced in developing safe, efficient physical data gathering agents.

In chapter 2, we begin by describing the decision-theoretic formulation of the safe, active data gathering problem in partially known environments, and examine how approximations are used in state of the art to address different parts of the problem. We will also examine some of the active data gathering systems that have been successfully deployed while highlighting the need for modeling motion constraints.

In chapter 3, we describe the algorithm to ensure the safety of a vehicle operating in partially known environments by using a library of emergency maneuvers, and evaluate its performance on multiple full-scale helicopters.

In chapter 4, we describe an algorithm that leverages emergency maneuvers to enable sensor motion planning for gathering for ensuring the safety of the vehicle. We evaluate and compare the performance of the algorithm on full-scale helicopters equipped with an actively controlled nodding laser.

In chapter 5, we will move from sensor motion planning to vehicle motion planning for active data gathering. We prove the need for reasoning about

Table 1.1: Mapping from challenges to chapters

high dimensional belief space dynamics for budgeted, multi-resolution data gathering problem, and present an online solver for the same problem in chapter 6.

Finally, we conclude with a summary of the work, and a discussion of some of the more interesting results. We discuss some of the unresolved problems of the resulting planning framework and remaining open questions.

2

Background

In this chapter we first introduce a decision-theoretic formulation for safe active data gathering in physical spaces, section 2.1. We then present approximations made by state of the art techniques to tractably address the active data gathering problem, section 2.2. We then discuss successfully deployed data gathering robotic systems and briefly present the need of the work presented in this document, section 2.3.

2.1 Decision Theoretic Formulation of Safe Data Gathering

Under the decision theoretic regime, autonomous data gathering agents integrate the actions and observations made by the robot to update robot's belief about the environment and select the action that maximizes a reward function in expectation. Let the combined state of the environment and the robot be $s \in \mathcal{S}$. Let the high dimensional state of the environment be given by $m \in \mathcal{S}$. If the state of the agent and the environment is known and the agent takes an action $a \in \mathcal{A}$ it causes the environment to transition from state s to state $s' \in \mathcal{S}$ with probability $\Omega(s, a, s') = P(s_{t+1} = s' | s_t = s, a_t = a)$, we use subscript to denote the time step. However, at any given time the agent has a probabilistic belief about its state and the state of the world, $b \in \mathcal{B}$ be the probability distribution over the state, $b : \mathcal{S} \rightarrow [0, 1]$. Let, \mathcal{A} be action space of the robot and the sensor. Let $\pi : \mathcal{B} \rightarrow \mathcal{A}$ be the policy that maps from belief to action space. Let $o \in \mathcal{O}$ be an observation and $Z(s', a, o) = P(o_{t+1} = o | s_{t+1} = s', a_t = a)$ be the observation model. The reward function, $R_B : \mathcal{A} \times \mathcal{B} \rightarrow \mathbb{R}$, is belief dependent and assigns rewards to belief, action pairs. The dynamics constraints on the robot and the sensors are given by $h(\pi(b_t), \pi(b_{t-1})) \leq 0$. The safety constraint is represented by $s_A(b_t, \pi(b_t)) \leq 0$. The cost of the trajectory is given by $\sum_{t=0:T} C(\pi(b_t))$. And the total budget for the information gathering mission be given by B .

$$\pi^* = \arg \max_{\pi \in \Pi} \sum_{t=0}^T \mathbb{E}_{b_t \sim P(b_t | \pi, t)} [R_B(b_t, \pi(b_t))] \quad (2.1)$$

$$\begin{aligned}
h(\pi(b_t), \pi(b_{t-1})) &\leq 0 \forall t = 0 : T \\
s_A(b_t, \pi(b_t)) &\leq 0 \\
b_{t+1}(s') &= \eta Z(s', a, o) \sum_{s \in \mathcal{S}} \Omega(s, a, s') b(s) \\
\sum_{t=0:T} C(\pi(b_t)) &\leq B
\end{aligned}$$

Where, η denotes a constant normalizer.

To solve equation 2.1 optimally, the solver has to operate in the belief state-space \mathcal{B} , which is continuous and has at least the same dimensionality as the environment. Moreover, it has to reason about the expectation over all possible future observations and trajectories. It is computationally infeasible to address the problem of active data gathering online in a decision-theoretic fashion. Adding motion and safety constraints to the problem further increase the complexity. Nonetheless, it is a significant problem to solve, and researchers have addressed the problem through various approximations as presented in the next section.

2.2 Related Work

Active Data Gathering

The problem of planning for data gathering is a well studied one and finds its roots in sequential hypothesis testing [Wald, 1945]. Sequential hypothesis testing have since been extended to account for mobile sensing within the framework of Bayesian reasoning [Cameron and Durrant-Whyte, 1990]. The work on active mobile sensing can be broadly classified into three paradigms, myopic approaches that ignore motion constraints, non-myopic approximate solvers that reason about the cost of moving or model motion constraints, and decision theoretic approaches.

Ignoring Motion Constraints - Sensor Motion Planning

Early work of J.J. Gibson [Gibson, 1950] proposed that perception is due to the combination of the environment in which the agents exists and how those agents interact with the environment. He noticed that optical flow observed by a mobile agent is controlled by its motion, hence the agent can actively control how and what it senses in the environment. One of the early works of active data gathering through sensor configuration planning on a mobile agent was presented in [Tenenbaum, 1970], it involved control of camera parameters for optimizing edge detection. In the seminal work [Bajcsy, 1988], a framework was proposed for one-step greedy active data gathering for camera control to minimize misclassification risk. [Ballard, 1990] suggested the concept of an animate vision system that can control the gaze of a low-resolution camera to achieve higher resolution imagery. These

algorithms focus on controlling the sensor motion or parameters to achieve better task performance.

As cameras became higher resolution and more computationally powerful, some of these techniques for active exposure and focus control got absorbed in onboard camera hardware. However, for Lidars, a sensor that still suffers from limited bandwidth, active sensor control approaches are still practically beneficial. Some work has addressed the need for controlling lidar to focus on specific targets to localize them, for example in [Arora et al., 2013] we focus a Lidar on the ship deck to enable better localization of it.

However, for safe data gathering in partially known environments, an agent needs to focus its sensor on gathering information to ensure vehicle safety as well as task-specific data. Such tasks present an added challenge of reasoning about agent's dynamic constraints to ensure its safety. We describe an algorithm to enable such data gathering in chapter 4. In the next subsection we present how myopic data gathering techniques are used for vehicle motion planning.

Ignoring Motion Constraints - Vehicle Motion Planning

Optimization of sensor configuration does not need to reason about path-length or budget constraints of the agent. Hence, myopic or heuristic driven approaches can provide efficient solutions. Similar approaches were first applied to mobile data gathering agents for controlling their trajectories to gather data. Yamauchi in his seminal work on frontier-based exploration [Yamauchi, 1997] suggested a myopic approach for information gathering. Later works have resulted in a variety of myopic (next-best-view) approaches that incorporate information theoretic measures for problems like object recognition [Denzler and Brown, 2002], mapping [Stachniss et al., 2005], and scene reconstruction [Chen et al., 2011]. While such algorithms have shown to be useful in their respective applications, they typically rely on restrictive assumptions on the representations, objective functions and do not have guarantees on global optimality.

Finite-horizon, model predictive control methods, [Bourgault et al., 2003, Ryan and Hedrick, 2010] provide an improvement over myopic techniques, but they do not have performance guarantees beyond the horizon depth and the run times to operate online in large state-spaces. Charrow et al. [Charrow et al., 2015] present a finite horizon, information theoretic approach where a set of sensing locations is identified, and the system travels to the most promising one.

The approaches discussed above do not model the budget constraint and assume that the agents have sufficient runtime to cover the entire environment. Leading to an assumption that coverage is equivalent to information gathering, [Plonski and Isler, 2016]. While such strategies are computationally efficient, they fail to account for the constraint on travelling distance effec-

tively. As a result, the computed routes can lead to oscillatory behavior. In the next sub-section, we discuss planning approaches that model budgetary constraints of agents.

Modelling Motion Constraints

The problem of planning routes to gain information is NP-hard [Krause, 2008]. Decision-theoretic approaches to information gathering have been improved over the years either by approximating solutions through sampling the action space [Pineau et al., 2003] or by restricting the space of problems to metric spaces, [Lim et al., 2016]. Chen et al. presented how POMDP-Lite [Chen et al., 2016] can be used to solve large information gathering problems in a relatively small time by augmenting the reward function with information theoretic rewards. Often, to reduce the run-time of these solvers, the maximum likelihood observations are used instead to define deterministic belief space dynamics [Hsiao et al., 2008]. Nonetheless, the computational requirements of these decision theoretic solvers render them unusable in an online setting, especially for MAVs, with limited computation carrying capabilities.

Another approach is to invoke a long horizon planner [Yu et al., 2014, CHEN et al., 2014, Zhang and Vorobeychik, 2016]. However, these approaches are far from real-time. The recursive greedy algorithm [Singh et al., 2009], Branch and bound [Binney et al., 2013] require computation exponential in the size of the problem instance due to the large blow-up in the search space with increasing budget. An alternative is to utilize a finite-horizon solver that solves the problem for only a portion of the budget at a time [Hollinger et al., 2009].

Given the run-times of these algorithms, there is a need for faster methods to compute informative routes. We present Randomized Anytime Orienteering (RAOr) an algorithm that can efficiently solve for routes that maximize correlated reward functions subject to constraints on route length in chapter 6.

Guaranteeing Safety

Substantial literature exists on ensuring safety for autonomous vehicles. We focus our attention on ensuring safety for static, uncertain environments.

Wikman et al. [Fraichard and Asama, 2004b] introduced the notion of *Inevitable Collision State* (ICS), a state for which irrespective of future vehicle trajectory, a collision is inevitable. The general approach to safety is to avoid ICS, while assuming unknown regions in the environment to be obstacles. Some of the early work for ground robots relied on making sure that vehicle can stop within sensor range while applying maximum longitudinal deceleration [Fox et al., 1997]. This technique is effective for vehicles that have the capability to apply large longitudinal decelerations. Similar approach to safety has been adopted by aerial vehicles as well

[Scherer et al., 2012c, Goerzen and Whalley, 2011, Adolf and Dittrich, 2012]. However, stopping distance based velocity limit does not exploit the complete dynamics of the vehicle, leading to conservative velocity limits.

Another paradigm is to simplify the non-linear dynamics of the UAVs and plan a path that is guaranteed to stay within the known unoccupied region. Mixed integer linear programming is used in [Schouwenaars et al., 2004] to plan paths that stay within the known region. Simplified dynamics model in a sampling based graph is used in [Frazzoli et al., 2002] while limiting the maximum planning time to ensure safety. The assumption is that the planner can always plan an obstacle free path if allowed to run until the maximum planning time. [Enright et al.] uses Dubin curves to plan paths within the known space. Simplified of dynamics, coupled with a reliance on a planner to generate a safe path online leads to conservative robot behavior that does not fully exploit sensory and dynamics capabilities of the vehicle.

We present an emergency maneuver library based method that utilizes the true dynamics of the vehicle to find a positive control invariant set in the known unoccupied space. We formulate the problem of generating this library as a NP hard path survivability optimization [Dey et al., 2011]. We prove the path diversity problem to be monotonic, sub-modular leading to an efficient, bounded sub-optimal algorithm [Streeter and Golovin, 2007] to generate the trajectory set. In the next section we discuss four exploration strategies that we consider as the state of the art.

Data Gathering Systems

There have been multiple information gathering systems with significant field results. We briefly describe relevant systems and their planning paradigms. Starting with the river exploration MAV by the Air lab at FRC, CMU [Jain et al., 2015]. The MAV used myopic frontier based planning, to follow a river and map its banks. The system worked in GPS-denied environments using visual odometry for state estimation. It primarily relied on using lidar for detecting water surface of the river and mapping its banks. Similar, sensor suite and data gathering methods were used by the bridge inspection MAV [Yoder and Scherer, 2016].

Infrastructure inspection often has hard constraints on the resolution at which the information needs to be gathered. Bircher et al. [Bircher et al., 2015] leverage such constraints and the model of infrastructure to be inspected to generate an optimized information gathering route. Given the limited sensing range or the sensing constraints, the information gathering planners in the above use cases do not need to reason about multi-resolution information gathering. Both these approaches assume they have enough battery to complete the mission and do not consider the flight time or budget constraint while gathering information.

[Wettergreen et al., 2014] use relatively low resolution satellite imagery

to guide a ground rover to efficiently collect science data. The on-board informative path planner uses a variant of a recursive greedy, generalized cost benefit algorithm [Zhang and Vorobeychik, 2016]. The methodology reasons about multi-resolution nature of data gathering while applying an approximate algorithm of planning for data gathering under budget constraints. However, the rover itself gathers data at a constant resolution. Jones et al. [Jones et al., 2014] use a similar one-step greedy approach to reason about multi-resolution data gathering for mapping craters on lunar surface. [Mascarich et al., 2018] uses randomly generated geometric trees, similar to [Hollinger], that respect budget and yaw-rate constraints to enable localization of radiation sources using UAV.

2.3 Developing Algorithms that Enable Safe Data Gatherers in Physical Spaces

Active data gathering through mobile robots has been an area of interest since the late 80's. Early approaches were limited to heuristic driven myopic policies due to limited computation available. As computing and sensing hardware improved, these approaches were applied to ground robots where the motion constraints and sensing constraints were not restrictive. Leading to next-best-view based myopic approaches being effective for ground-based exploration applications. With UAV systems maturing over the last decade, there is a need for developing algorithms that enable safe, multi-resolution data gathering systems that do not rely on myopic, greedy methods to overcome oscillatory behavior and lead to efficient, multi-resolution information gathering under budgetary and sensory constraints.

In this work we describe algorithms to solve three different aspects of the safe, efficient data gathering namely enabling safe operations of mobile agents in partially known environments while utilizing their dynamic limits, reasoning about pertinent data gathering for the safety of the mobile agents while modeling their dynamic constraints and reasoning about task-specific, multi-resolution data gathering with budget constraints. Throughout this work, we assume that the agent is operating in a partially known environment while the agent's pose is known.

We start by describing an algorithm that enables safe operations of mobile agents in partially known environments while fully exploiting vehicle dynamics and known space available in the next chapter 3.

3

Ensuring Safety in Partially Known Environments

Active data gathering in physical environments and applications like cargo delivery, surveillance, people transport, reconnaissance etc. require the robots to operate in unstructured, partially known environments at high speeds. The robots should ensure safety while navigating in such environments without compromising on performance. A popular method to guarantee safety relies on limiting the vehicle speed such that it can come to a stop using longitudinal deceleration within the known obstacle-free volume [Buchberger et al., 1993]. This method fails to fully exploit either the vehicle's dynamics or known unoccupied volume, leading to unsatisfactory performance. Another method includes planning a trajectory such that its initial part takes the robot towards the goal while it ends in a control invariant set that lies within the known obstacle-free region [Schouwenaars et al., 2004]. Although this method fully exploits the known space it limits the planning horizon. In this chapter, we examine the problem of ensuring safety for mobile autonomous systems while maintaining the capability to operate the vehicle at its performance limits. The key idea is to ensure that the vehicle is always in a safe state from which it can transition to a loiter pattern or come to a stop within the known obstacle-free space. All these states are inside the control invariant set of the robot [Blachini, 1999], which is a well-known approach to ensure feasibility for model predictive control applications [Michalska and Mayne, 1993]. Determining loiter patterns or trajectories resulting in complete stop in various environments is computationally challenging especially when the robot has non-linear dynamics. Additionally, it is required that the safety evaluation has a low worst-case response time so that it can be used for on-line motion planning at high speeds.

In order to ensure the on-line capability of the safety evaluation, the problem is decoupled in an off-line and an on-line part. The off-line part generates an optimized set of trajectories enabling the robot to reach a safe state and stay within the known obstacle free region for an infinite time horizon. The trajectory set is designed to maximize the probability of finding at least one emergency maneuver in the known unoccupied environment. The on-line part determines if the set contains a collision-free trajectory

regarding the current state and environment of the robot. Thereby, the off-line generated trajectory set reduces the search space for the on-line part. This safety evaluation approach serves as a computationally tractable algorithm with bounded run time.

It can be shown, that the problem of generating this optimized trajectory set is NP-hard [Branicky et al., 2008]. We present an efficient, bounded sub-optimal approximate solution that finds a trajectory set maximizing the probability of containing at least one safe trajectory given a prior obstacle distribution. The proposed novel safety assessment approach is based on an emergency maneuver library and is compared to stopping distance. The proposed approach was evaluated through a variety of experiments conducted on multiple rotorcraft. In all evaluated scenarios the novel safety assessment approach outperforms known common approaches by enabling higher safe velocities of the rotorcraft while guaranteeing safety for the rotorcraft at all times.

The main contributions discussed in this chapter are as follows:

- *Generation of an emergency maneuver library that allows for on-line safety assessment of robotic systems at high speeds in unknown environments*
- *Efficient algorithm for the library generation with bounded sub-optimality*
- *Experimental evaluation from field tests with an autonomous full-sized helicopter flying at speeds of 56m/s*

3.1 Safety Definition

Autonomous mobile robots have matured over the years. As these systems are developed for field applications [Scherer et al., 2012c, Abdel-Malek et al., 2006, S.Q. Marlow, 2009, Enright et al.], the need for robust and safe autonomous robots is highlighted. Previous work on safety of autonomous robots can be broadly divided into two paradigms. One of the paradigms is to make sure that the vehicle can stop within the sensor range while applying maximum allowed longitudinal deceleration [Scherer et al., 2012c, Goerzen and Whalley, 2011, Adolf and Dittrich, 2012]. The stopping distance based velocity limit does not exploit the complete dynamics of the vehicle, leading to conservative velocity limits.

Another paradigm is to simplify the non-linear dynamics of the UAVs and plan a path that is guaranteed to stay within the known unoccupied region. Mixed integer linear programming is used in [Schouwenaars et al., 2004] to plan paths that stay within the known region. Simplified dynamics in a sampling based graph is used in [Frazzoli et al., 2002] while limiting the maximum planning time to ensure safety. The assumption is that the planner can always plan an obstacle free path if allowed to run until the maximum

planning time. [Enright et al.] uses Dubins curves to plan paths within the known space. These methods also suffer from not being able to exploit the vehicles full dynamic capabilities.

It is also important to quantify the safety of the mobile autonomous systems. [Mettler et al., 2010] suggested using distance from obstacles as a metric for safety of a robot navigating through an obstacle field. This metric does not take into account the sensory and dynamic limitations of the vehicle, thus it cannot ensure safety of the vehicle.

We present a safety metric that considers both sensory and dynamics constraints of the vehicle to evaluate the vehicle safety. We then present an emergency maneuver library based method that utilizes the true dynamics of the vehicle to find a positive control invariant set in the known unoccupied space. We formulate the problem of finding this library as a NP hard path diversity optimization [Green, Colin J. and Kelly, 2011, Branicky et al., 2008, Erickson and LaValle, 2009, Dey et al., 2011]. We prove the path diversity problem to be monotonic, sub-modular leading to an efficient, bounded sub-optimal algorithm [Streeter and Golovin, 2007, Krause and Guestrin, 2007] to generate the trajectory set.

The safety of a mobile autonomous system is dependent on its sensory and dynamic capabilities. In a fully-known environment a mobile system is unsafe if it enters a state for which there exists no trajectory that avoids a collision, such a state is called an Inevitable Collision State [Fraichard and Asama, 2004a]. In a static partially-known environment the unknown regions may contain obstacles. Therefore, to ensure the safety of the mobile robot its state should be constrained such that it can always transition to a terminal feasible invariant set [Schouwenaars et al., 2004] that enables the robot to stay within the known obstacle-free volume for an infinite time horizon. We now formally define safety for robots operating in uncertain environments. Let, $\mathbf{x}(t)$ be the state of the robot at time t in the state space \mathcal{X} which is in a manifold $\mathcal{X} \subset \mathcal{S} \in \mathbb{R}^n$. The workspace of the robot is defined as \mathcal{W} and the occupancy of the robot system in the workspace at a certain state is given as $\mathcal{V}_A(\mathbf{x}(t)) \subset \mathcal{W}$. The known space in the workspace at a given time t is denoted as $\mathcal{K}_t \subset \mathcal{W}$. The occupancy of the known obstacles at time t is given by $\mathcal{O}_t^{obs} \subset \mathcal{K}_t \subset \mathcal{W}$. Let $\Phi_F(\mathbf{x})$ be the search space of trajectories for a given state \mathbf{x} , that end in a terminal feasible invariant set. Let $\phi(\mathbf{x})$ be such a trajectory and let $\phi(\mathbf{x}, \tau)$ be the state of the vehicle at time τ , along the trajectory $\phi(\mathbf{x})$, which is by definition rooted at state \mathbf{x} . Then any trajectory followed by the vehicle can be considered safe if for all states on the trajectory there exists a trajectory $\phi(\mathbf{x})$ which completely lies inside the known obstacle-free space at that time. Equation (3.1) presents this definition formally:

Definition 1 (Motion Safety):

$$\forall t, \forall \tau, \exists \phi(\mathbf{x}) : \mathcal{V}_A(\phi(\mathbf{x}, \tau)) \subset (\mathcal{K}_t \setminus \mathcal{O}_t^{obs}) \quad (3.1)$$

A trajectory is considered safe till time T if for all $t \leq T$ equation 3.1 is satisfied. In the next section, we discuss how this safety definition can be enforced on mobile autonomous vehicles in real-time with the use of an emergency maneuver library.

3.2 Approach

Finding a trajectory that satisfies (3.1) online is non-trivial due to computation costs involved, especially if the robot's dynamics are non-linear. State of the art methods approximate vehicle dynamics or use stopping distance based safety criterion, which leads to robots performing well below their dynamics and sensory capabilities. We provide a method that enables for guaranteeing the safety of the vehicle in static environments, while exploiting the limits of vehicle dynamics and considering the available known obstacle-free space. We split the problem into two parts, first, we efficiently compute a reduced set of terminal invariant trajectories offline then use this reduced set to search for safe terminal invariant trajectories online. In the next section we discuss how to compute the terminal invariant trajectory set resulting in the emergency maneuver library. Before discussing how this emergency maneuver library can be used online to ensure safety of the mobile robot.

Offline Generation of a Library of Emergency Maneuvers

Instead of solving for dynamically feasible trajectories that end in a terminal invariant set on-line, we approximate the search space by a finite set of such trajectories. The trajectory set is designed such that the probability that at least one of the trajectories stays collision-free, given a prior on the obstacle configuration for a given state is maximized for a static environment. We show that this problem is NP hard [Branicky et al., 2008, Erickson and LaValle, 2009] and then prove that it is monotonic sub-modular, providing a sub-optimality bound for a greedy algorithm.

Modeling Safety as Survivability

We want to find a set of trajectories that are control invariant and maximize the probability that the set contains at least one collision-free trajectory. The trajectory set is optimized for a spatial stochastic process defined in $r \in \mathcal{W}$, that captures the distribution of obstacle configurations that the robot is likely to encounter during its lifetime. We assume this spatial field is given by $\zeta(u, r)$, where u is the event of point r being unoccupied/free. The probability density function defined by this process is then given by $p_\zeta(u, r)$. Probability that there is no obstacle inside a volume $V \subset \mathcal{W}$ is thus given $P_u(V) = \int_V p_\zeta(u, r) dr$. The volume which is swept by the

robot following a certain trajectory ϕ is expressed as $V(\mathcal{V}_A(\phi)) = \{\mathbf{r} | \mathbf{r} \in \mathcal{W}, \exists \tau \mathbf{r} \in \mathcal{V}_A(\phi(\tau))\}$, [Abdel-Malek et al., 2006]. We use the shorter notation $V_\phi = V(\mathcal{V}_A(\phi))$ to denote the volume swept by the robot. The probability of a path being safe is given by the probability of the swept volume being free of obstacles.

$$P_u(\phi) = P_u(V_\phi)$$

which allows to determine the probability of path ϕ_1 or ϕ_2 being unoccupied

$$P_u(\phi_1 \cup \phi_2) = P_u(\phi_1) + P_u(\phi_2) - P_u(\phi_1 \cap \phi_2)$$

where,

$$P_u(\phi_1 \cap \phi_2) = P_u(V_{\phi_1} \cap V_{\phi_2}).$$

Using the inclusion-exclusion principle, we get the probability of a path set $\Phi = \{\phi_1, \phi_2, \dots, \phi_n\}$ having an obstacle free path as

$$P_u(\Phi) = \sum_{k=1}^n (-1)^{k-1} \left(\sum_{1 \leq i_1 \leq \dots \leq i_k \leq n} P_u(\phi_{i_1} \cap \dots \cap \phi_{i_k}) \right). \quad (3.2)$$

In order to maximize the safety of the robot, a finite path set Φ must be determined maximizing the probability that at least one path is collision-free. This is formulated as the path diversity problem:

Problem 1 (Maximizing Path-Set Survivability): *The desired trajectory set Φ_d maximizes the probability of finding at least one obstacle-free path.*

$$\begin{aligned} \Phi_d &:= \arg \max P_u(\Phi) \\ &\text{subject to } \|\Phi_d\| < N_\Phi \\ &\text{where, } \Phi_d \subseteq \Phi \subseteq \Phi_F \end{aligned} \quad (3.3)$$

Φ_F is the search space of trajectories. Since the path diversity problem is known to be NP-hard, we present a greedy method to optimize (3.3). But before that we prove that greedily optimizing equation (3.3) is bounded sub-optimal. To prove bounded sub-optimality we prove the $P_u(\Phi)$ is sub-modular and monotonically increasing in the cardinality of Φ .

Monotonicity Proof

In this section we show that the probability of at least one path in a path set being collision-free is monotonically increasing with the cardinality of the path set.

Proposition 1 (Monotonicity of Path Sets). *Given a path set Φ_A , a path ϕ_a and a path set $\Phi_B = \{\Phi_A, \{\phi_a\}\}$ the probability that the set Φ_B contains at least one collision-free path is bigger or equal than for the set Φ_A*

$$P_u(\Phi_B) - P_u(\Phi_A) \geq 0.$$

Proof. The probability of the path set Φ_b having at least one obstacle-free path is given by

$$P_u(\Phi_B) = P_u(\Phi_A \cup \{\phi_a\}).$$

Using inclusion exclusion principle lead to

$$\begin{aligned} P_u(\Phi_B) &= P_u(\Phi_A) + P_u(\phi_a) - P_u(\Phi_A \cap \phi_a) \\ P_u(\Phi_B) - P_u(\Phi_A) &= P_u(\phi_a) - P_u(\Phi_A \cap \phi_a). \end{aligned} \quad (3.4)$$

For all ϕ_a , $P_u(\phi_a) \geq 0$ and $\max[P_u(\Phi_A \cap \phi_a)] = P_u(\phi_a)$, which is the case when all the volume covered by path ϕ_a is already covered by Φ_A . This implies that

$$P_u(\phi_a) - P_u(\Phi_A \cap \phi_a) \geq 0$$

and inserted in (3.4) leads to $P_u(\Phi_B) - P_u(\Phi_A) \geq 0$ ■

Prop. 1 shows that adding more trajectories to a path set, cannot decrease the probability of finding an obstacle-free path in the set. In other words, $P_u(\Phi)$ is monotonically increasing in the cardinality Φ .

Sub-Modularity Proof

In order to show that a greedy algorithm for the path diversity problem (Prob. 1) is bounded sub-optimal, we will show that $P_u(\Phi)$ is a sub-modular set function. This means, that the difference in the probability P_u that a single trajectory makes when added to the path set decreases as the size of the path set increases.

Proposition 2. *Let there be a path set $\Phi_\Gamma \subseteq \Phi_Y \subseteq V$, where $P_u : 2^V \rightarrow \mathbb{R}$. Now, assume a path ϕ_e , such that $\phi_e \subseteq V \setminus Y$. Define $\Phi_{\Gamma+e} = \{\Phi_\Gamma, \phi_e\}$, $\Phi_{Y+e} = \{\Phi_Y, \phi_e\}$. For sub-modularity*

$$\Delta(e|Y) < \Delta(e|\Gamma)$$

where, $\Delta(\cdot)$ is the discrete derivative.

Proof. The discrete derivative of $\Delta(e|\Gamma)$ is defined as

$$\begin{aligned} \Delta(e|\Gamma) &= P_u(\Phi_{\Gamma+e}) - P_u(\Phi_\Gamma) \\ &= P_u(\Phi_\Gamma \cup \phi_e) - P_u(\Phi_\Gamma) \\ &= P_u(\Phi_\Gamma) + P_u(\phi_e) - P_u(\Phi_\Gamma \cap \phi_e) - P_u(\Phi_\Gamma) \\ &= P_u(\phi_e) - P_u(\Phi_\Gamma \cap \phi_e) \end{aligned}$$

and similarly

$$\Delta(e|Y) = P_u(\phi_e) - P_u(\Phi_Y \cap \phi_e).$$

Taking the difference of the discrete derivatives

$$\begin{aligned}
\Delta(e|\Gamma) - \Delta(e|Y) &= \\
&= P_u(\phi_e) - P_u(\Phi_\Gamma \cap \phi_e) - P_u(\phi_e) + P_u(\Phi_Y \cap \phi_e) \\
&= P_u(\Phi_Y \cap \phi_e) - P_u(\Phi_\Gamma \cap \phi_e) \\
&= P_u((\Phi_\Gamma \cup \Phi_{Y/\Gamma}) \cap \phi_e) - P_u(\Phi_\Gamma \cap \phi_e) \\
&= P_u((\Phi_\Gamma \cap \phi_e) \cup (\Phi_{Y/\Gamma} \cap \phi_e)) - P_u(\Phi_\Gamma \cap \phi_e) \\
&= P_u(\Phi_\Gamma \cap \phi_e) + P_u(\Phi_{Y/\Gamma} \cap \phi_e) \\
&\quad - P_u(\Phi_\Gamma \cap \phi_e \cap \Phi_{Y/\Gamma}) - P_u(\Phi_\Gamma \cap \phi_e) \\
&= P_u(\Phi_{Y/\Gamma} \cap \phi_e) - P_u(\Phi_\Gamma \cap \phi_e \cap \Phi_{Y/\Gamma})
\end{aligned}$$

Applying Baye's Rule

$$P_u(\Phi_\Gamma \cap \phi_e \cap \Phi_{Y/\Gamma}) = P_u(\Phi_\Gamma | (\phi_e \cap \Phi_{Y/\Gamma})) P_u(\Phi_{Y/\Gamma} \cap \phi_e)$$

it follows that

$$\Delta(e|\Gamma) - \Delta(e|Y) = P_u(\Phi_{Y/\Gamma} \cap \phi_e)(1 - P_u(\Phi_\Gamma | \phi_e \cap \Phi_{Y/\Gamma})).$$

With $P_u(\Phi_{Y/\Gamma} \cap \phi_e)(1 - P_u(\Phi_\Gamma | \phi_e \cap \Phi_{Y/\Gamma})) \geq 0$ the equation can be rewritten as

$$\Delta(e|\Gamma) - \Delta(e|Y) \geq 0.$$

■

Greedy Algorithm for Generating Emergency Maneuvers

Since, $P_u(\Phi)$ is monotonic sub-modular, the path diversity problem (Prop. 1) can be greedily optimized while maintaining a sub-optimality bound of $(1 - 1/e) \approx 63\%$ [Nemhauser et al., 1978, Krause and Guestrin, 2007]. We describe the greedy algorithm in Alg. 1. We start with an empty trajectory

Algorithm 1: Greedy Optimization for a Emergency Maneuver Trajectory Set

Initialize: $\Phi_G = \emptyset$

```

while  $|\Phi_G| < N_{CE}$  do
   $\phi_s = \arg \max_{\phi \in \Phi_F / \Phi_G} P_u(\Phi_G \cup \{\phi\})$ 
   $\Phi_G = \{\Phi_G \cup \{\phi_s\}\}$ 
end

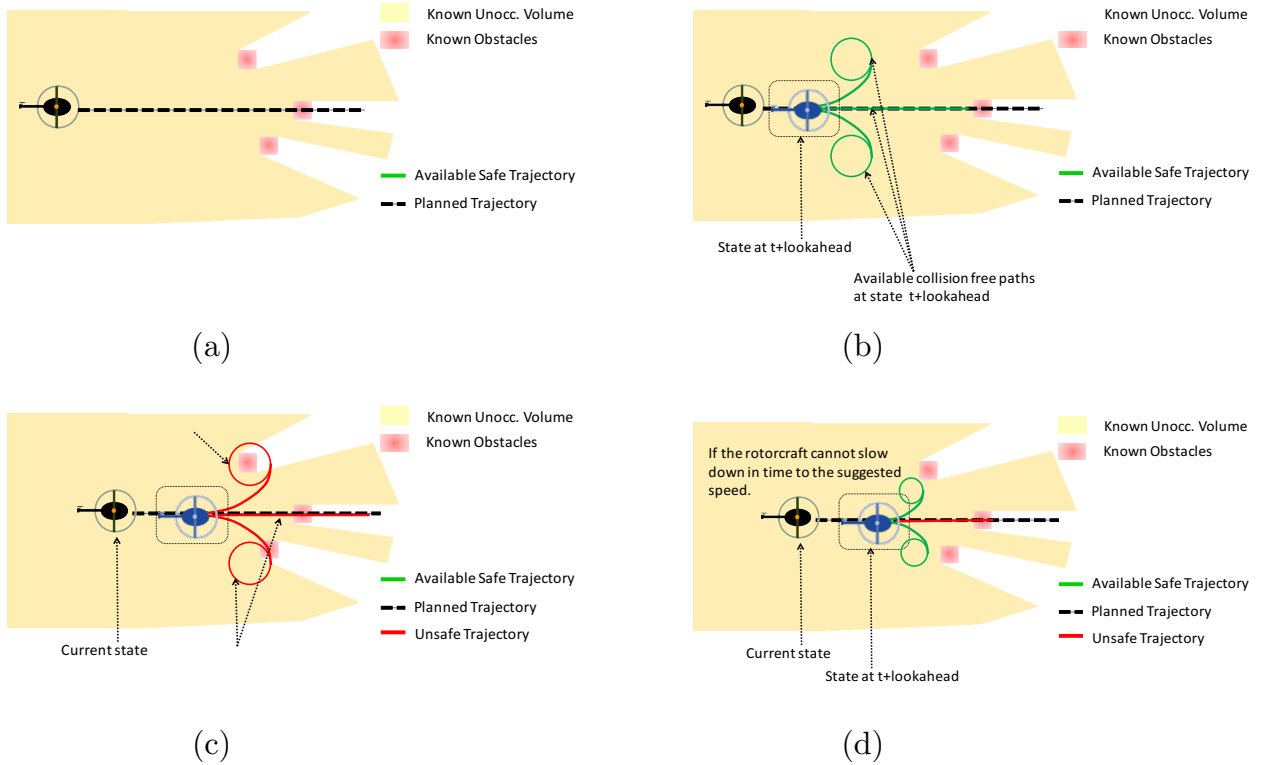
```

set and search through Φ_F to find the trajectory that maximizes P_u . This trajectory is saved in Φ_G and in the next step, the search for trajectory that maximizes P_u is conducted in Φ_F / Φ_G , and added to Φ_G . The process of greedily selecting trajectories from Φ_F / Φ_G and adding them to Φ_G is repeated till the desired number of trajectories N_{CE} have been added. In the

next section we explain how to use this greedily generated set to guarantee safety.

Using Emergency Maneuvers to Ensure Safety

We ensure the safety of the mobile autonomous system by using the emergency maneuver library to enforce the constraint that the current and next state of the system always lies in the positive invariant set, which does not intersect the obstacles and stays within the known volume.



The algorithm to ensure safety is explained in Alg. 2 and demonstrated in Fig. 3.1. Let $\sigma : [0, T] \rightarrow \mathcal{X}$ be the nominal trajectory that the vehicle is following to reach the goal. Let, Φ_{S_t} be the emergency maneuver library for state at time t in σ and Δt be the time interval between safety checks.

The algorithm queries the emergency maneuver library at a future state of the system, and ensures it can transition to an emergency maneuver which lies in known obstacle free space. If there are no such maneuvers, one of the emergency maneuvers computed at the previous step (for the current state) are executed. Otherwise the vehicle carries on its nominal trajectory. This algorithm has a fixed maximum response time and is guaranteed to keep the vehicle safe.

Figure 3.1: Safety Algorithm: (a) A motion planner sends a planned trajectory to the safety algorithm (b) The safety checker queries an emergency maneuver library to ensure maneuvers exist from the trajectory that lie entirely in known free space. (c) Eventually the safety checker encounters a case where the library is unable to find a maneuver for the trajectory (d) The safety checker tries to slow down the vehicle until the library is successful. If not, the safety checker reverts to the previous guaranteed safe trajectory.

Algorithm 2: Emergency Maneuver Trajectory Set Application for Reactive Safety

```

Initialize:  $t = 0$ 
 $\Phi_{\text{Previous}} = \Phi_{S_t}$ 

while mission active do
   $\Phi_{\text{New}} = \{\emptyset\}$ 
  for  $\forall \phi_c \in \Phi(s_{t+\Delta t})$  do
    if  $\forall \tau \mathcal{V}_{\mathcal{A}}(\phi_c(\tau)) \subseteq (\mathcal{K}_t \setminus \mathcal{O}_t^{\text{obs}})$  then
       $\Phi_{\text{New}} = \{\Phi_{\text{New}}, \{\phi_c\}\}$ 
    end
  end
  if  $\Phi_{\text{New}} = \{\emptyset\}$  then
    | execute  $\phi_e \in \Phi_{\text{Previous}}$ 
  else
    |  $\Phi_{\text{Previous}} = \Phi_{\text{New}}$ 
    | follow  $\sigma$ 
  end
end

```

3.3 Results

Reduction in Sensing Requirements

We generated the emergency maneuver library to ensure the safety of three full-scale autonomous helicopters, equipped with a large field of view range sensor. The dynamic constraints of the helicopter are given in Tab. 3.1.

Constraint	Velocity $\ v(t)\ $	
	$\geq 20 \text{ m/s}$	$< 20 \text{ m/s}$
Roll [$^\circ$]	25.00	28.50
Roll rate [$^\circ/\text{s}$]	15.00	—
Heading rate [$^\circ/\text{s}$]	—	28.50
Longitudinal vel. [m/s]	60.00	20.00
Vertical vel. [m/s]	5.00	5.00
Longitudinal accel. [m/s^2]	0.75	0.75
Vertical accel. [m/s^2]	1.00	1.00

Table 3.1: Constraints on trajectory

Given these constraints we approximate Φ_F , by five hundred trajectories each forming a positive control invariant set. The trajectories for this application end in a hover and can trivially be extended to end in a loiter if desired. Each trajectory slows down the helicopter using the maximum allowed deceleration. The trajectories are generated by sampling the roll rate and z acceleration uniformly. Once the helicopter has made a 180° coordinated turn the radius of the turn is fixed and the vertical velocity is forced to be 0 m/s . We use a constant resolution three dimensional grid as our representation and

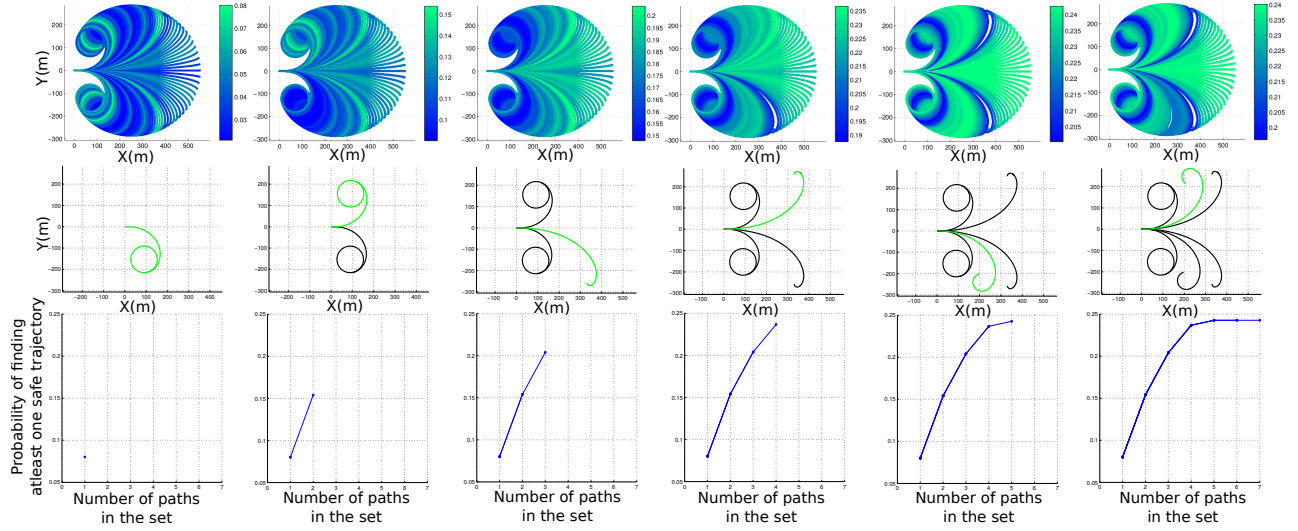


Figure 3.2: Generation of emergency maneuver library for one state. From left to right the plots step through the generation of emergency maneuver library for 6 iterations. The top row displays the search space from which the current trajectory is picked, where each trajectory is colored according to the probability of not passing through an obstacle in the set. The middle row shows the greedily selected maneuver in the current step in green and existing maneuvers in the set in black. The bottom row shows the total probability of finding at least one maneuver in the set not passing through an obstacle. The robot starts at 25m/s longitudinal velocity for all the maneuvers and for illustration purposes, is restricted to move in the xy plane. The benefit of adding new maneuvers diminishes as more trajectories are added and almost levels off after 5 trajectories.

assume uniform probability of occupation of each voxel. The probability of a trajectory set containing at least one unoccupied trajectory is calculated using inclusion-exclusion principle as suggested in [Branicky et al., 2008]. Fig. 3.2 steps through the emergency maneuver library generation process for the robot motion restricted to a plane starting at 25m/s forward longitudinal velocity. The probability of at least one maneuver in the set surviving reduces with each trajectory being added and almost levels off at about 5 trajectories. Given a trajectory set we can calculate the sensor range required for different velocities. Given an emergency maneuver library, the minimum sensor range required for a certain velocity is calculated as

$$\text{range} = \min_{\phi_c} (\max(\zeta(\phi_c))). \quad (3.5)$$

The function ζ returns a vector of the euclidean distances between starting state \mathbf{x} and all the states in $\phi_c \in \Phi_G(\mathbf{x})$.

The best case sensor range required while using the emergency maneuver library is given by (3.5). The worst case is the same as the stopping distance. Hence, the emergency maneuver library is guaranteed to provide at least as much performance as using only the stopping distance for the safety evaluation. In Fig. 3.3 the different requirements on the sensor range for stopping distance and the emergency maneuver library are illustrated.

Case Study 3.1: Safety Execution Test

To ensure that the rotorcraft can execute safety maneuvers and keep the vehicle safe, safety check was initially executed on the rotorcraft in a scenario with simulated obstacle. Fig. 3.4 shows an early test of the safety checker and eml. In the next section we show how the EML was able to keep the vehicle

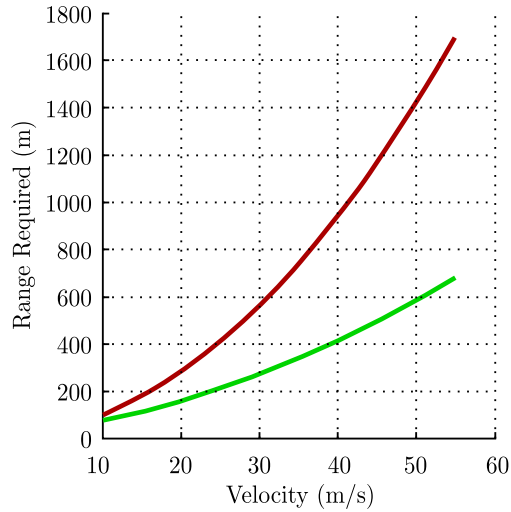


Figure 3.3: Changes in Sensor Requirements. The sensor range required for safe operation of the vehicle when using stopping distance for safety is displayed in red, in green is the sensor range required for safe operation of the vehicle when using emergency maneuver library for helicopter safety.

from entering unknown spaces in adverse condition like sensor outages.

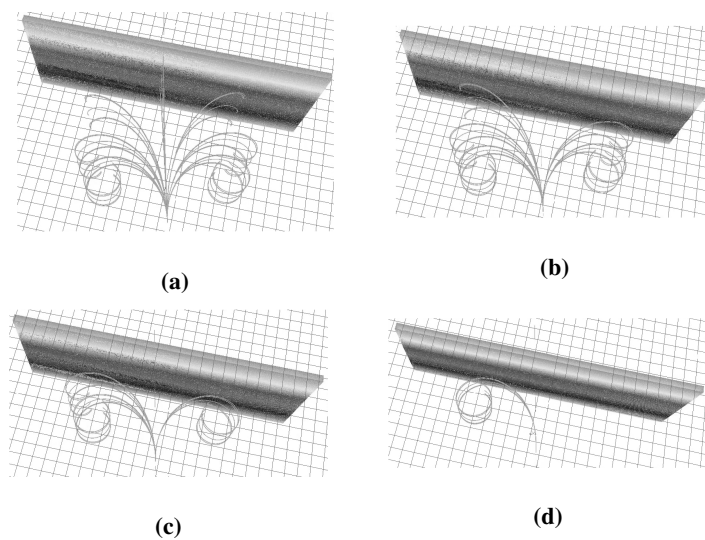


Figure 3.4: Data from a flight test conducted on 18th December 2013 in Manassas, Virginia. a) Helicopter approaches a large simulated wall with the emergency trajectory libraries with no emergency maneuver in contact with the wall. b) As the helicopter gets closer to the wall, the emergency maneuvers intersect the wall and become invalid. Only valid maneuvers are displayed. c) More emergency maneuvers are pruned away as they come in contact with the wall d) An emergency maneuver is executed as the future state is no longer safe.

Case Study 3.2: Sensor Failure

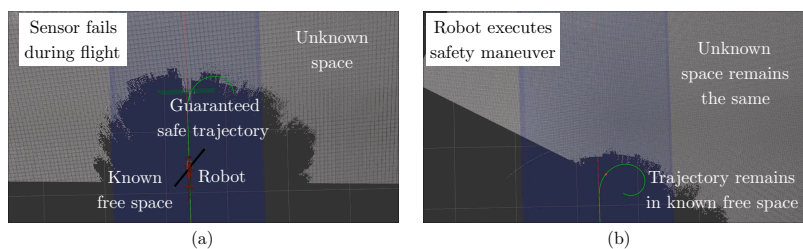


Figure 3.5: Guaranteeing safety even when the sensor fails (a) A situation where the sensor has failed. In such a situation, the space that is unknown will remain unchanged due to lack of updates from the sensor. However, the current trajectory being followed is guaranteed safe by construction as it always lies in the space that is known to be free (b) The robot eventually executes the evasive maneuver as the unknown space remains unchanged. This ensures the robot perpetually remains in a safe state.

Fig. 3.5 shows the vehicle safely surviving during sensor failure. Note that the current trajectory being followed is guaranteed safe as it lies in the known free space. In the subsequent time-steps, even though the trajectory planning algorithm continues to plan new paths, the motion planner does not update the trajectory as the unknown space remains unchanged. The robot continues to follow this trajectory and eventually executes the evasive maneuver section as shown in Fig. 3.5. Hence, the robot perpetually remains in the known free space ensuring the system remains safe till a human operator can intervene.

Case Study 3.3: Quantico and Mesa Flights

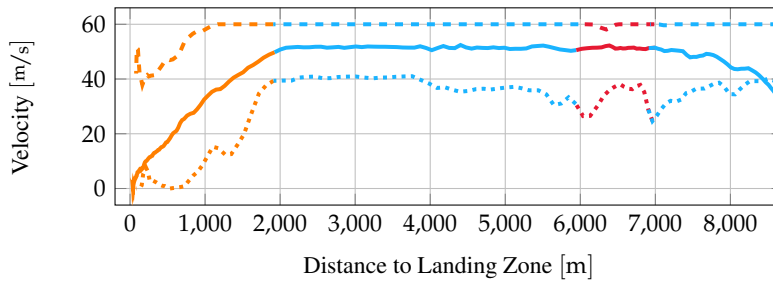
We can quantify the performance of an emergency maneuver library by calculating the maximum safe velocity it enables the helicopter to fly at and the planning time it allows the planner before it becomes imperative for the helicopter to execute the emergency maneuver library. Fig. 3.6 shows the maximum safe velocity and allowed planning times for a flight test conducted in Quantico, Virginia. The red line shows the path where the helicopter is turning towards the landing zone. The orange part of the path corresponds to the part of the mission for which the sensor on the helicopter focuses on the landing zone for its evaluation. This implies, when the helicopter is moving through the path in orange the sensor stops looking for obstacles and the helicopter comes increasingly close to the known/unknown volume boundary, leading to a drop in maximum safe velocity and allowed planning time. The red part of the path corresponds to turns, it should be noted how the maximum safe velocity according to the stopping distance decreases as the vehicle turns. This happens due to a reduction in effective range of the sensor because of the sparsity of observations in front of the vehicle while turning. The maximum safe speed by the emergency maneuver library is unaffected, as it efficiently utilizes the known space.

Fig. 3.8 shows the maximum safe velocity and allowed planning times for seven flight tests conducted in Quantico, Virginia which are shown in Fig. 3.7.

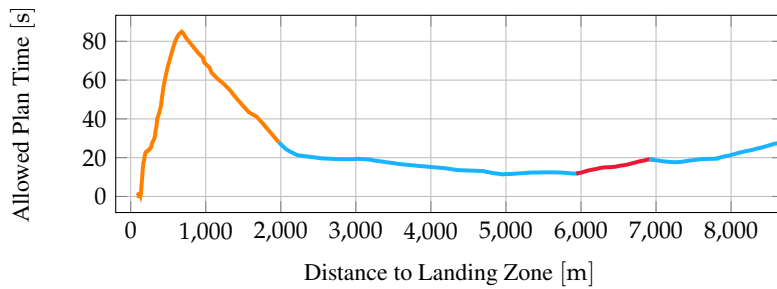
As can be seen in Fig. 3.8, the maximum safe speed is always greater than the helicopter speed, which means the helicopter is always safe. Furthermore, the stopping distance based safe velocity limit is always considerably below the executed velocity which shows that the emergency maneuver library approach is less conservative than the stopping distance approach. The use of the emergency maneuver library also enable higher available planning times leading to a better overall performance of the motion planning approach due to longer available computation time.



(a)

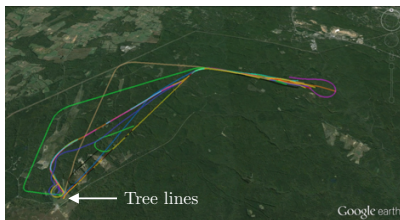


(b)

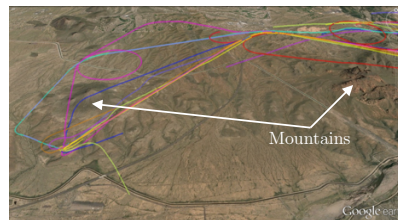


(c)

Figure 3.6: Safety Quantization: Flight test in Quantico, Virginia. a) Shows an autonomous landing mission conducted in Quantico, Virginia on Unmanned Little Bird. b) Shows the safe velocity of the helicopter with the emergency maneuver library during the flight tests in dashed line, the executed velocity in solid line and the safe velocity if stopping distance is used in dotted line. c) This figure shows planning time available to the planner, before the vehicle will reach the edge of known space and execute one of the emergency maneuvers. The planning time calculated assuming the helicopter will follow the current planned trajectory.



(a)



(b)

Figure 3.7: Different planning problem distributions in different environments. The lines denote the path traced by the helicopter superimposed on a satellite image (a) Missions in Quantico, VA, where obstacles are towers, trees and no fly zones (b) Missions in Mesa, AZ, where obstacles are mountains and no fly zones.

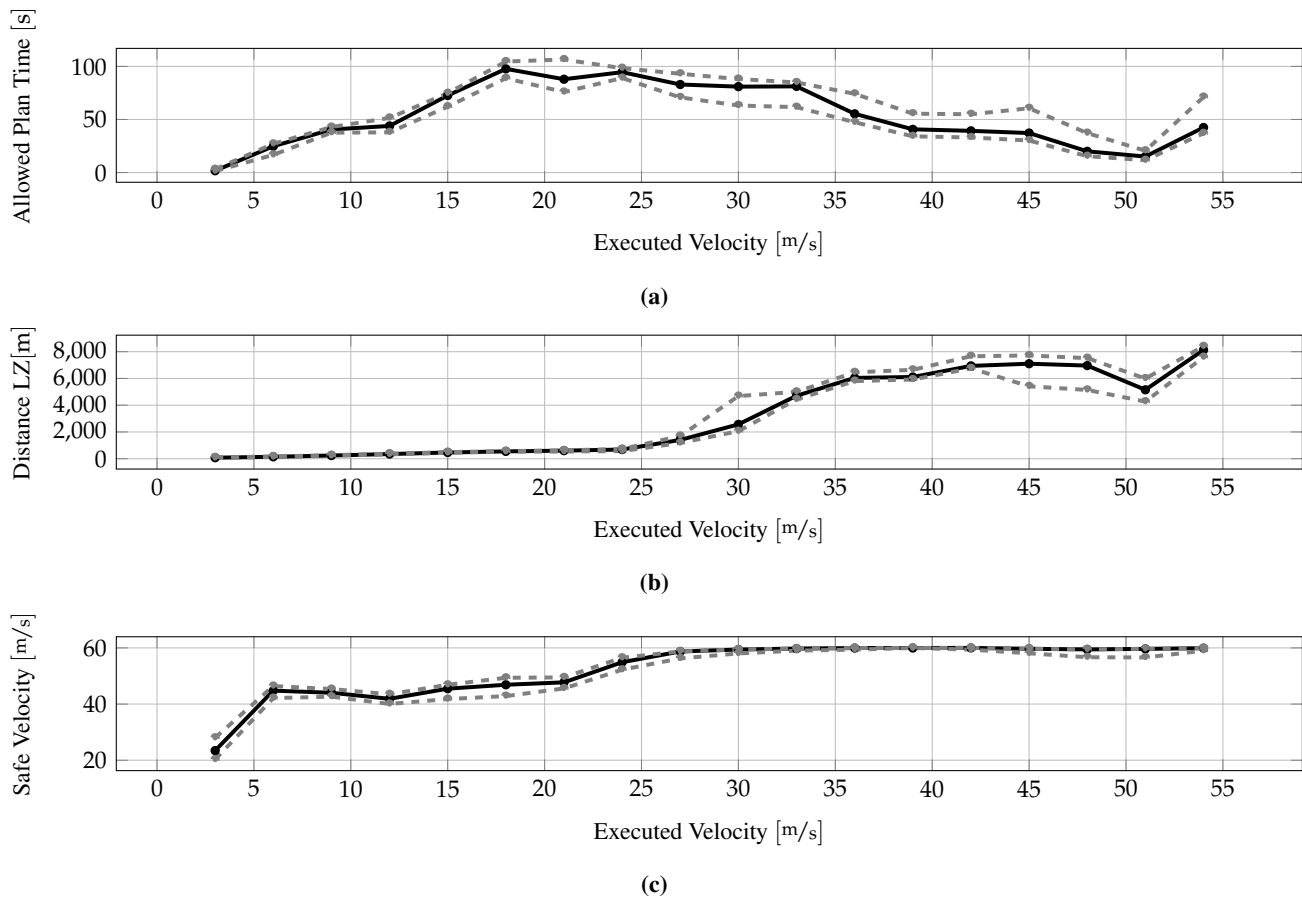


Figure 3.8: The figures show the allowed plan time, distance to the landing zone (LZ) and, the safe velocity relative to the executed velocity of the helicopter. The black line shows the mean and the gray dashed line illustrates the upper and the lower bound of the measurements of all considered flight tests.

3.4 Summary

In this chapter, we presented development and evaluation of emergency maneuver library to guarantee the safety of high speed mobile autonomous systems, online in unknown environments. The algorithm determines the maximum velocity for which safety can be ensured, given a future path of the robot. Therefore, it takes into account the constraints of the perception system as well as the dynamics of the rotorcraft. Main contributions of the work are as follows -

- *Near-Optimal construction of emergency maneuver library* - We prove that constructing a safety library to maximize survivability is an NP-hard, sub-modular, monotonic problem, Prob. 1. Resulting in a near-optimal greedy algorithm for library construction, Alg. 1.
- *Online safety algorithm for non-linear vehicle dynamics* - Off-line generated trajectories enable real-time evaluation of the safety of the rotorcraft while fully exploiting vehicle dynamics. Enabling rotorcraft to fly safely at speeds of up to 60 m/s, while stopping distance based approach enables a

maximum speed of 40 m/s, Fig. 3.6.

- *More than 100kms of autonomous flight tests, over the course last three years-* Vehicle safety was ensured while operating in diverse environments and even in adversarial conditions like sensor outages.

Contributions towards Enabling Safe Data Gathering in Physical Spaces

This work formally defined the safety constraint (equation 3.1) in the decision-theoretic formulation (equation 2.1) of active data gathering in physical spaces. We also described an algorithm that enables enforcement of the safety constraint online, irrespective of the dynamics of the vehicle. The safety definition and the algorithm to ensure safety is valid for static, partially known environments with no pose uncertainty of the vehicle.

Having defined safety, in the next chapter we present an approach to enable active data gathering for the implicit need for a vehicle to be safe and for explicit mission requirements through sensor motion planning.

4

Sensor Planning for Implicit and Explicit Data Gathering

In the last chapter we described an algorithm to enable safe operation of mobile data gathering agents in partially known environments. In this chapter we use that algorithm to define reachability and optimize sensor motion to actively gather data for implicit requirement of safety. We also briefly describe an algorithm for active data gathering for the explicit mission-specific requirements.

Navigation through partially known environments, localization, mapping and manipulation of objects etc. are all tasks for which the robot is expected to detect objects, free space or features in the environment. The rate and accuracy of detection of information of interest dictate the performance of the robotic system. For example - we observed in the previous chapter that the speed at which a safe autonomous vehicle can operate is a function of the known space. Sensors to detect such features of the environment are often limited by their properties like field of view (FOV), resolution, sampling rate and signal to noise ratios. These sensors are either fixed [Huang et al., 2011, Furgale and Barfoot, 2010, Bills et al., 2011] with respect to the vehicle or in some cases move in constant pre-computed patterns [Scherer et al., 2012b,a]. The capabilities of such sensors and the robotic system can be augmented by actively controlling the sensor configuration to minimize the cost of completing the task assigned to the robot.

The problem of active perception is a well studied one and finds its roots in sequential hypothesis testing [Wald, 1945]. Active perception and adaptive sampling problems have since been extended to account for mobile sensing within the framework of Bayesian reasoning [Cameron and Durrant-Whyte, 1990]. Later works have resulted in various solutions that incorporate information theoretic measures for problems like object recognition, mapping, and scene reconstruction [Chen et al., 2011]. Gradient based methods for next best view and belief space planning have improved [Bourgault et al., 2002, Van Den Berg et al., 2012]. While such algorithms have shown to be useful in their respective applications, they typically rely on restrictive assumptions on the representations, objective functions and do not have guarantees on global optimality. Finite-horizon model predictive

control methods [Bourgault et al., 2003, Ryan and Hedrick, 2010] provide improvement over myopic techniques, but they do not have performance guarantees beyond the horizon depth and the run times to operate online in large state-spaces. POMDP based solvers [Myers and Williams, 2010], suffer from the same curse of dimensionality. The recursive greedy algorithm [Singh et al., 2009], Branch and bound [Binney et al., 2013] use the budget to restrict the search space. But require computation exponential in the size of the problem.

We overcome the curse of dimensionality by searching for a policy in reduced search space to actively control the sensor configuration. The policy function is learnt such that it maximizes the gain of information that is important for the completion of the mission of the robot while ensuring its safety as it navigates through unknown environments. We implement the policy on multiple full-scale autonomous helicopters equipped with an actively controlled nodding Lidar, using occupancy grid map as a world representation. The policy optimizes the use of sensor bandwidth to enable safe, high speed navigation and fast detection of landing zones. The main contributions of the chapter are as follows:

- *Computational complexity analysis for calculating expected information gain for a range sensor and an occupancy grid map representation.*
- *Policy function and feature design to enable online active sensor control that helps keep vehicle safe at high speeds.*
- *Efficient algorithm to approximate expected information gain for a range sensor detecting obstacles.*
- *Evaluation of the learnt policy on an autonomous full sized helicopter demonstrating guaranteed safe autonomous navigation at speeds of 56m/s and straight in approaches to landing zones, eliminating the need to hover.*

The task of enabling safe, high speed flight enforces tight temporal constraints on the sensor controller motivating the reactive, online approach. In the next section we formally setup the problem, section 4.2 presents with the approach overview. Section 4.2 and section 4.2 cover the implementation details including construction and the process of learning the policy parameters. Results of evaluation of the performance of the algorithm are discussed in section 4.4.

4.1 De-Coupling Sensor and Vehicle Motion Planning

In this section we formulate the problem as that of minimization of the expected cost of traversal from initial to goal state, through optimization of sensor trajectory. This problem is then shown to be the same as maximizing the gain of information that minimizes the cost of traversal. Information



Figure 4.1: Application Scenario: Top Left – Test Vehicle, Boeing Unmanned Littlebird. Top Right – Near Earth Autonomy M3 sensor suite, equipped with a actively controllable high range laser. Bottom – Example mission scenario, the vehicle is suppose to autonomously navigate to a pre-defined landing zone at high speeds, evaluate it and decide to whether to land or not while ensuring safety.

gain maximization problem formulation motivates and guides our approach, described in section 4.2.

Let the robot's state space be $\mathcal{X} \subset \mathbb{R}^n$. Let $\sigma : [0, T] \rightarrow \mathcal{X}$ be the state space trajectory and $C : \mathcal{X} \rightarrow \mathbb{R}$ be the cost function, where T is the time horizon. The boundary values are $\sigma(0) = \sigma_0$ and $\sigma(T) = \sigma_f$. Let the dynamics constraints on the robot be given by $h(\sigma(t), \dot{\sigma}(t), \ddot{\sigma}(t)) \leq 0$. The cost of the trajectory in a fully deterministic environment is $\int_0^T C(\sigma(t)) dt$.

Operating in partially known, unstructured environments the robot has to decide on its next action based on its current belief. Let the state space of the world be $\mathcal{W} \subset \mathbb{R}^m$, where world includes the uncertainty of the robot about its environment and it's pose. Let belief $b \in \mathcal{B}$, be a probability distribution over the state of the world, $b : \mathcal{W} \rightarrow \{0, 1\}$. Let's assume the belief of the robot at the start of the mission is b_0 . The belief of the robot changes as observations are made using a sensor. Let $\pi : \mathcal{B} \rightarrow \mathcal{A}$ be the policy that maps from belief to action space of the vehicle, then $\sigma(t) = \sigma_0 + \int_0^t \pi(b_t) dt$. Let the sensor's state space be $\mathcal{S} \subset \mathbb{R}^s$. Let $\sigma_s : [0, T] \rightarrow \mathcal{S}$ be the sensor trajectory. Let $\pi_s : \mathcal{B} \rightarrow \mathcal{A}_s$ be the policy that maps from belief to action space of the sensor, then $\sigma_s(t) = \sigma_{s0} + \int_0^t \pi_s(b_t) dt$.

The cost function changes as the belief of the robot evolves. To highlight this fact we represent the cost function as $C_b : \mathcal{X} \times \mathcal{W} \rightarrow \mathbb{R}$. Due to the stochasticity of the belief of the robot, it can only reason about expected cost of its policy, $\mathbb{E}_{p(b|\pi, \pi_s, b_0)} C_b(\cdot)$. The distribution of belief trajectories $p(b|\sigma, \sigma_s, b_0)$ at any given time is dependent on both state and sensor trajectory. Let the dynamics constraint on the motion of the sensor be given by $h_s(\sigma_s(t), \dot{\sigma}_s(t), \ddot{\sigma}_s(t)) \leq 0$. The full optimization problem can then be

defined as follows.

$$\arg \min_{\pi, \pi_s} \int_0^T \mathbb{E}_{p(b|\pi, \pi_s, b_0)} C_b(\pi(b(t)), \pi_s(b(t)), b(t)) dt \quad (4.1)$$

$$\begin{aligned} h(\sigma(t), \dot{\sigma}(t), \ddot{\sigma}(t)) &\leq 0 \\ h_s(\sigma_s(t), \dot{\sigma}_s(t), \ddot{\sigma}_s(t)) &\leq 0 \end{aligned}$$

In this chapter we study the problem of optimizing the sensor trajectory given a fixed vehicle trajectory $\sigma(\cdot)$. The problem is then reduced to.

$$\arg \min_{\pi_s} \int_0^T \mathbb{E}_{p(b|\sigma, \pi_s, b_0)} C_b(\sigma(t), \pi_s(b(t)), b(t)) dt \quad (4.2)$$

The constraints on the sensor actuation specified in eq. 4.1 also apply to eq. 4.2. Notice that, in this formulation the sensor motion can only result in gaining of information about the environment. Therefore, the minimization of the cost function with the π_s as the only variable, results in sensor gaining information that is important to minimize the required cost function. We call this information contextually important information, as it is important to sense this information to reduce the cost function. Let $M(x, \sigma(t), b(t)) \in \{0, 1\}$ return 1 if the information at point $x \in (W)$ in the world is contextually important and 0 otherwise, for a given belief $b(t)$, vehicle state $\sigma(t)$. Let $\mathbb{IG}(x, \sigma(t), \pi_s(b(t)), b(t))$ be the expected information gain at $x \in \mathcal{W}$ and time t .

Proposition 3 (Maximizing information gain to minimize trajectory cost). *Given a vehicle trajectory $\sigma(\cdot)$, optimizing sensor policy to minimize expected cost of the vehicle trajectory, eq. 4.2, is equivalent to optimizing sensor's policy for maximizing contextually important information gain, eq. 4.3.¹*

$$\arg \max_{\pi_s} \int_0^T \int_{\forall x \in \mathcal{X}} \mathbb{E}_{p(b|\sigma, \pi_s, b_0)} M(x, \sigma(t), b(t)) \mathbb{IG}(x, \sigma(t), \pi_s(b(t)), b(t)) dx dt \quad (4.3)$$

In our application, the robot is a full scale autonomous helicopter, the sensor is a nodding lidar, with an actively controlled pitch axis. The trajectory $\sigma_s(t)$ of the nodding sensor relative to the helicopter can be completely defined by its pitch angle profile in time $\rho(t)$, laser configuration ρ_c , which consists laser's fast axis resolution ρ_f and its pulse rate, ρ_p . The pose of the robot is given by a high accuracy GPS/INS system, the world representation is an occupancy grid map [Thrun, 2003]. The information gain evaluation for an occupancy grid map is presented in detail in the section 4.6. $M(x, \sigma(t), b(t))$, the contextual importance function is discussed in detail in section 4.2. In the next section we present an overview of our approach to solve eq. 4.3 for a mobile autonomous robot.

¹ The proofs of all the propositions are in section 4.6

4.2 Approach

Information gain based path planning is an NP-hard problem [Singh et al., 2009]. Moreover, the calculation of information gain itself is computationally expensive, see proposition 4. As a result, the computational cost involved in calculating information gain for a large number of rays over a large range, makes it non-trivial to conduct gradient based optimizations.

Proposition 4 (Complexity of calculating information gain). *For a laser sensory action with n_r rays, interacting with N cells of an occupancy grid map, the worst case computation complexity of calculating expected information gain is $O(3^{n_r}N)$.*

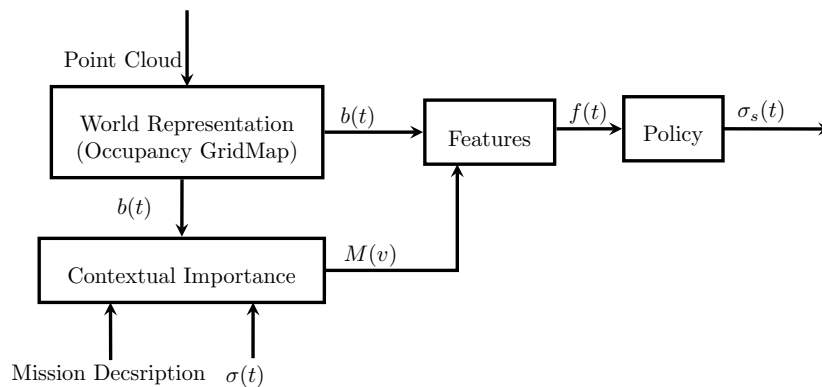


Figure 4.2: Approach Overview: The sensor is actively controlled through a policy that takes in the features that are extracted from robot’s belief. The features take expected information gain and contextual importance function into account.

To overcome the computational complexity issues and solve the problem online, we propose learning a reactive policy function that maps from features to action, (see Fig. 4.2). The policy learnt, is designed to maximize the contextual information gain in the next time step. For the application considered in the chapter, the policy learnt keeps the robot safe at high speeds and enables quick landing zone evaluation, enabling lower mission times. The data flow in the block diagram (Fig. 4.2) is explained in the rest of this section to provide an intuitive understanding of the suggested approach and its application.

The point cloud generated by the laser is used by the perception representation to form the robot’s belief about the world, we use an occupancy grid map as the world representation. The belief of the robot, along with the mission description and the intended robot trajectory are used to infer the contextual importance function. The contextual importance function presents the locations from which it is important for the robot to gain information to complete its task safely. The construction of this function is described in section 4.2. The contextual importance function along with the belief of the robot are used to calculate features. The policy function maps these features to sensory actions. The features are the only input to the policy. Therefore,

they should capture the saliences of the interaction between sensory actions, the representation and the environment. The construction of one such set of features is described in section 4.2.

The policy takes the features as input and provides the trajectory for the sensor, $\sigma_s(\cdot)$. For the application considered in the chapter, the policy is designed a for high range laser, nodding in the pitch axis with respect to the vehicle. The construction of the policy is described in section 4.2. The parameters of the constructed policy function need to be learnt to maximize the contextually important information gain. The procedure to learn the policy parameters is presented in section 4.3. In the next section we describe the construction of the contextual importance function for our application.

Contextual Importance Function

Finding the contextual importance function is as hard as solving for the original optimization problem, described in section 4.6. We approximate this function by defining $M(x, \sigma(t), b(t)) = 1$, for all x , that might be important to the vehicle given the belief $b(t)$ and the robot state $\sigma(t)$. In this work, we consider the structure of the contextual importance function to have two components - safety $M_{safe}(\cdot)$ and landing zone evaluation $M_{lz}(\cdot)$.

Pertinent Information for Implicit Data Gathering

We use the emergency maneuver library, as described in chapter 3, to enforce the safety constraint. If there exists a trajectory in the emergency maneuver library that starts from the current state of the vehicle and stays within the known obstacle free region for infinite time, the vehicle can be considered safe. Therefore, it is important to sense the volume occupied by emergency maneuver trajectories corresponding to the planned trajectory.

Given a vehicle state $\sigma(t)$ at time t , let there be a function $\kappa(x, \sigma(t), b(t)) \in \{0, 1\}$ which is one for the points inside the volume occupied by emergency maneuver library for state $\sigma(t)$ if it is unsafe and zero otherwise. The contextual importance function for safety is then given as.

$$M_{safe}(x, \sigma(t), b(t)) = \kappa(x, \sigma(t), b(t)) \quad (4.4)$$

Pertinent Information for Explicit Data Gathering

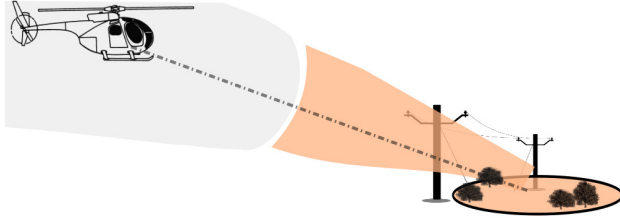
The mission objective of the helicopter is to navigate safely from start to goal and then land. Therefore, it is important to evaluate landing zones before the helicopter commits to a landing. The landing zone is represented as a 2D grid, with each cell storing the probability of that cell being a safe landing site. Let, the landing zone grid be represented by \mathbb{LZ}_G . Each cell inside the landing zone is allocated contextual importance of one.

$$M_{lz}(x, \sigma(t), b(t)) = 1 \forall x \in \mathbb{LZ}_G \quad (4.5)$$

The combined contextual importance function of safety and landing zone evaluation is given as

$$M(\cdot) = \max(M_{safe}(\cdot), M_{Lz}(\cdot)) \quad (4.6)$$

Fig. 4.3, illustrates the contextual importance function for a helicopter



navigating towards a landing zone to land.

Policy Parametrization for Sensor Planning

We have discussed that the problem of directly optimizing the sensor trajectory to minimize the cost function is NP hard and the cost function evaluation is computationally expensive, making local optimization impractical. Therefore, we develop a mapping from features to nodding actions that can be used online. We introduce the features and policy for controlling the nodding of the laser for ensuring vehicle safety in sections 4.2 and 4.3. The policy to evaluate landing zone is described in section 4.3.

The policy function is re-defined as, $\pi_s : \mathbb{R}^f \rightarrow \mathbb{R}^a$, where \mathbb{R}^f is the feature space and \mathbb{R}^a is the action space. The policy is developed such that it can guarantee the safety of the vehicle at high speeds if the environment allows so. We restrict the action space to constant velocity nods to make the search computationally tractable. We learn a policy offline that maximizes the information gain in the worst case scenario. The worst case scenario and the process of learning the policy function is presented in section 4.3.

We want to optimize the use of sensor bandwidth and avoid wasting it looking at regions that are already known, inaccessible or unimportant to the mission. Therefore the features have to cover visibility, information gain and contextual importance function. In the next subsection we cover the feature design and then present the policy function.

Input Features for Sensor Planning Policy

To calculate the relevant field of view for the sensor, we compute and store the contextually important expected information gain along a grid of pitch (θ) and yaw (ϕ) directions, with rays originating from laser's center. This forms a 2D map $\zeta_m(\theta, \phi) \rightarrow \mathbb{R}$, mapping view direction to contextually weighted expected information gain. We generate this map by tracing rays through 3D

Figure 4.3: Contextual Importance Function – The figure illustrates the region with contextual importance function greater than 1 in orange, and region which is known to the robot in grey. As the robot navigates, the known region has no more information to be gained that might affect its future actions. The volume around the trajectory, bounded by the emergency maneuver library for the future unsafe states and volume inside the landing zone is contextually important. It is for gaining this information that we need to optimize the sensory actions.

occupancy grid representation and calculating contextually weighed expected information gain along the rays. The algorithm for calculating the expected information gain along the rays is presented in [Julian et al., 2014]. Average range at which information is gained r_e is also used as a feature for the policy. r_e can be calculated while calculating ζ_m . The detailed algorithms to calculate ζ_m and r_e are presented in detail in section 4.6. The policy function can then be represented as eq. 4.7 below.

$$[\dot{\rho}, \rho_f, \rho_{max}, \rho_{min}] = \pi(\zeta_m, r_e) \quad (4.7)$$

The FOV to scan ρ_{max}, ρ_{min} , is inferred using ζ_m . The ρ_{max} is given by maximum pitch in ζ_m for which the information gain is greater than 0. The ρ_{min} is given by minimum pitch in ζ_m for which the expected information gain is greater than 0. The speed of the nod is stored as lookup table against r_e at which the information to be sensed. We now look at how we learn the mapping (lookup table) between r_e and the nodding velocity that maximizes the information gain for that range. The nodding actions are limited to time $t_r = 1.4s$, which is the same as the maximum time taken by representation to update.

4.3 Policy Search

We want to find a nodding speed at which to scan the volume to maximize the gain of contextually important information. The standard method to find the optimal policy parameters is to run the system either in simulation or in field and search the parameter space for values that minimize the expected cost over a range of scenarios. But given we are designing a policy to keep the vehicle safe in all conditions, we search for the optimal policy for the worst case scenario. In the worst case scenario, the time available to the sensor for scanning is minimal and the uncertainty to be reduced about the presence of obstacles is maximum. In other words the information to be gained is maximum. Subsection 4.3 describes how information gain is related to the object detection uncertainty. Section 4.3 develops the worst case scenario for a guaranteed safe autonomous mobile robot navigating through unknown environments. This scenario is used to learn the nodding actions that maximize contextually important information gain. Where, information gain is a function of reduction in uncertainty about the presence of obstacle of interest. In the case of a rotorcraft the smallest obstacles of interest are wires. If we can ensure the detection of wires, all other obstacles like trees, building and hills are guaranteed to be detected. Section 4.3 describes the algorithms to efficiently calculate probability of detection given sensory actions, enabling efficient search of policy parameter ($\dot{\rho}$).

Worst Case Policy Search Scenario for Implicit Data Gathering

The average range from which contextually weighted information is to be gained is given by r_e . Let $V_s : X \rightarrow \mathcal{R}^3$ be the function that returns the volume bounded by the emergency maneuver library at a given state. In the worst case scenario the sensor has minimum time to scan the maximum volume. Therefore we assume the vehicle is unsafe at $\sigma(t + t_r)$, where t is the current time. We also assume V_s is monotonic in speed of the input state and $V_s(\sigma_1) \in V_s(\sigma_2)$ if speed of the vehicle at σ_1 is less than the speed at σ_2 , while pose at σ_1, σ_2 is the same. Assuming we are operating in an unknown environment.

Proposition 5 (Worst Case Scenario for Safety). *The sensor would have to reduce uncertainty for maximum volume in the environment if the vehicle is moving at the maximum speed that enables it to stay safe at r_e .*

To learn the nodding speed for a given r_e , we assume the vehicle is at state σ_{safe} , where the speed at σ_{safe} is the maximum speed such that at least one of the emergency maneuvers lie within a sphere of range r_e . To guarantee the safety of the vehicle in the worst case scenario, the sensor will have to reduce the uncertainty about the existence of obstacles in $V_s(\sigma_{safe})$. We use this worst case scenario to learn the policy parameters $[\dot{\rho}, \rho_c]$ for a given r_e . Next section explains the relationship of uncertainty reduction or expected information gain and the probability of detection of objects given sensory actions. Information gain as probability of detection of worst case obstacle Let the probability that the smallest obstacle against which the autonomous system has to guarantee safety is present at x is given by $p_x(o)$. For brevity we use $p(o)$ instead of $p_x(o)$. Let the probability of detection of obstacle given there exists an obstacle at x be $p(d|o)$ and probability of detecting an obstacle given there is no obstacle at x is given by $p(d|o')$. We can express the expected information gain given $p_x(o)$ as a function of $p(d|o)$ and $p(d|o')$ (see section 4.6).

Proposition 6 (Monotonicity of Expected Information Gain). *Assuming there are no false positives, $p(d|o') = 0, p(d'|o') = 1$. For a given $p(o)$, \mathbb{IG} can be completely expressed as a function of $p(d|o)$ and monotonically increases in $p(d|o)$.*

Fig. 4.4 illustrates proposition 6.

The no false positives assumption holds for a lidar capable of processing multiple returns. To find the effectiveness of an action we evaluate $p(d|o)$ for a given $p(o)$ and calculate the information gain in the mission relevant region. The action that provides the maximum information gain is selected as policy for given features. In the next section we present an efficient algorithm to calculate $p(d|o)$, that allow the creation of policy lookup table tractably. Efficient calculation of probability of detection We describe a method that enables us to calculate expected information gain for a range

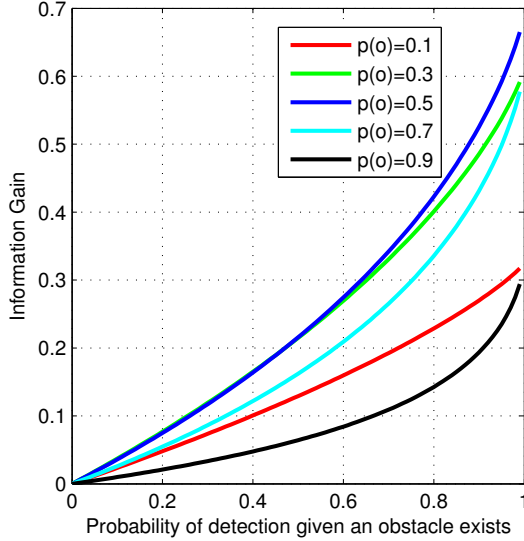


Figure 4.4: Information Gain – The expected information gain given a prior ($p(o)$) is monotonic in probability of detecting the obstacle if the obstacle is present, $p(d|o)$. Assuming there are no false positives, $p(d|o') = 0$. This implies for maximizing information gain, one may maximize $p(d|o)$

sensor efficiently. To compute expected information gain we calculate $p(d|o)$ for a given $p(o)$. The naive method of calculating $p(d|o)$ is through sampling configurations of the object and computing ray intersections with the object given the sensor nodding motion. This algorithm has the complexity of $O(RM)$, where R is the total number of rays generated by the laser and M is the number of samples drawn from $p(o)$. If an object exists in $6D$ space, and for each dimension we sample N particles. The computational complexity is then $O(RN^6)$. We introduce an algorithm that changes this computational complexity to $O(R^3N^4)$ by reasoning about ray-object intersections in the object's c -space in spherical coordinates. We further exploit the symmetry and structure of the problem of sensing for safety to reduce the complexity to $O(R_l^3N)$, where R_l is a small fraction of the actual number of rays.

The algorithm for efficient calculation of $P(d|o)$ is presented in Alg. 3. The input to the algorithm is sensor trajectory (σ_s), partial pose ($x_{partial} \in \mathbb{R}XSO(3)$) of the object of interest. The partial pose consists of the range of the object from the laser and its relative orientation. We assume that the size of the object is small compared to the query range, which results in negligible change in projection of the object to $[\theta, \phi]$ plane of the spherical coordinate frame, centered at the laser. The algorithm returns a function $p(d|\theta, \phi, x_{partial})$ that provides the probability of detection of object if its in $x_{partial}$ configuration at any $[\theta, \phi]$. We assume the object is detected if n hits are detected on the object. It is also assumed that if the ray and the object intersect, the hit will be reported with a probability of p_h .

Each laser beam, is projected as an ellipse on $[\theta, \phi]$ plane given its beam width. We approximate this ellipse as a convex polygon in $[\theta, \phi]$ plane. The projection of the object of interest to the same plane is also approximated as

a convex polygon. To compute probability of detection of the object we need to calculate intersections between the object of interest and laser rays. We compute these intersections by projecting laser beam polygons space to the c-space of the object of interest in the (θ, ϕ) plane.

The $Intersect(\cdot)$ function (Alg. 4), calculates the intersecting regions for the set of c-space polygons amongst themselves and stores them as a set $Y_{int}^{S_{partial}}$. It also returns the list of members of its input set that intersected to form a polygonal region $i \in Y_{int}^{S_{partial}}, \forall i \in Y_{int}^{S_{partial}}$ as $\lambda_{int}^i \in \Lambda_{int}$. The output of Alg. 4 is used by $Probability(\cdot)$ function (Alg. 5), to calculate the probability of detection given an object at $x_{partial} \forall [\theta, \phi]$.

Algorithm 3: *EfficientCalculationofP(d|o)*

Input : $\sigma_s, x_{partial}$

Output : $p(d|\theta, \phi, x_{partial})$

$S_o \leftarrow ProjSpherical(s_{partial})$

$S_L \leftarrow GeneratePolygons(\sigma_s)$

$S_{L,o} = S_L \oplus S_o$

// The minkowski sum projects polygons in

// S_L to the space of S_o .

$[Y_{int}, \Lambda_{int}] \leftarrow Intersect(S_{L,o})$

$p(d|\theta, \phi, s_{partial}) \leftarrow DetectionProbability(Y_{int}, \Lambda_{int})$

Return : $p(d|\theta, \phi, s_{partial})$

Proposition 7 (Probability of Detection vs. Distance). *Given a fixed angular motion profile of a lidar sensor, if an object is detected when n laser hits are reported on it, and the probability of an intersecting ray reporting a hit is p_h . The probability of detection of the object decreases with increase in the distance between the object and the lidar sensor, assuming the relative orientation of the object and lidar is constant.*

Since we are reasoning about safety, leveraging prop. 7, we maximize the information gain by evaluating $p(d|o)$ at the furthest range we are interested in, where the $p(d|o)$ is minimum. In our case that range is given by r_e . This reduces the task to maximizing the information gain on a 2D spherical manifold $[\theta, \phi]$ at range r_e . So the configuration space of the object restricted to a 2D plane is 3D. Assuming, we use N particles for each dimension to approximate $p(o)$, the computational cost of evaluating an action is given by $O(R^3N)$.

We restricted the action space to constant velocity nodes, hence the laser points lie at a constant distance from each other in pitch and yaw space, forming a grid. This symmetry can be exploited by finding $p(d|o)$ for every configuration of the object in a single grid cell as $p(d|o)$ is the same for

Algorithm 4: *Intersect(.)*

Input : $S_{L,o}$ **Output** : $Y_{int}^{S_{partial}}, \Lambda_{int}$ **Initialize**: $Y_{int}^{S_{partial}} = S_{L,o}, \lambda_{int}^i = i \forall i \in Y_{int}^{S_{partial}}, Y_{prev} = S_{L,o}$ **while** $|Y_{prev}| > 1$ **do** $Y_{new} = NULL$ **for** $j \in Y_{prev}$ **do** **for** $k \in Y_{prev} - [1 : j]$ **do** $Y_{new} = Y_{new} \cup (j \cap k)$ $\lambda_{int}^{end+1} = \lambda_{int}^j \cup \lambda_{int}^k$ **end** **end** $Y_{int}^{S_{partial}} = Remove(Y_{int}^{S_{partial}}, Y_{new})$ $Y_{int}^{S_{partial}} = Y_{int}^{S_{partial}} \cup Y_{new}$ $Y_{prev} = Y_{new}$ **end****Return** : $Y_{int}^{S_{partial}}, \Lambda_{int}$

Algorithm 5: *Probability(.)*

Input : $Y_{int}^{S_{partial}}, \Lambda_{int}^{S_{partial}}, \phi, \theta$ **Output**: $p(d|\phi, \theta, s_{partial})$ $i \leftarrow ReturnPolygonContaining(\theta, \phi, Y_{int}^{S_{partial}})$ $p(d|\phi, \theta, s_{partial}) = \sum_{k=n}^{|\lambda_{int}^i|} \binom{|\lambda_{int}^i|}{k} p_h^k (1 - p_h)^{|\lambda_{int}^i| - k}$ **Return** : $p(d|\phi, \theta, s_{partial})$

every cell. Also, the intersection of the object need only be checked with the points in l^2 radius of the grid cell, where l is the length of maximum side of the bounding box of the object of interest. Let these number of points in l^2 radius be R_l . The complexity of evaluating an action is $O(R_l^3 N)$. For a flying helicopter the smallest obstacle of interest are wires, we focus our analysis on $10m \times 0.07m$ wires, where $0.7m$ is the width of a standard high tension wire on tall electric towers. We pessimistically assume that the wire is only $10m$ long. A typical nodding action may have $R_l = 11$, we use $N = 10000$, the reduction in computation time is by a factor of 10^{15} , see section 4.6.

Alg. 3 is evaluated exhaustively for the complete nodding velocity space at finite discrete intervals, to create the lookup table of range vs nodding velocity. The slowest nodding speed is restricted such that one complete nod occurs in maximum $t_r = 1.4s$. The action providing maximum information gain is stored as the nodding time for that range. Fig 4.5 presents plots for expected probability of detection of a wire, given a uniform distribution of wires.

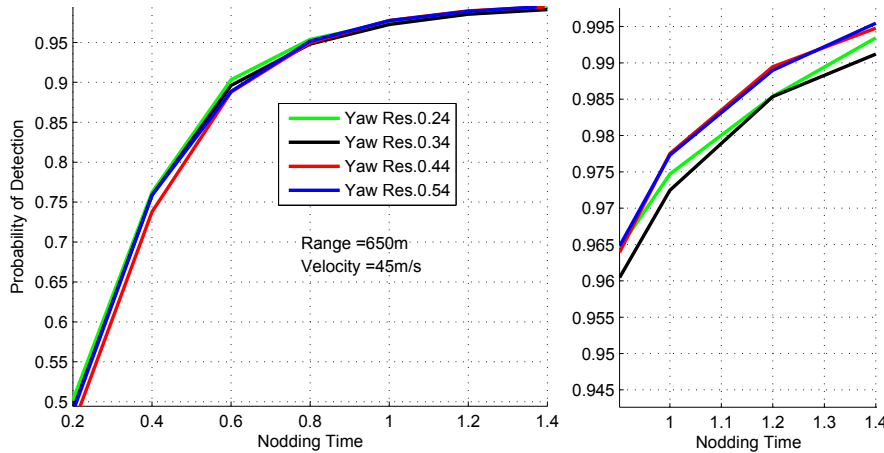


Figure 4.5: Expected Probability of Detection: The expected probability of detection of a wire if it exists, $\mathbb{E}_{p(o)} p(d|o)$ for a given action is an indicator of how good an action is. The plots show $\mathbb{E}_{p(o)} p(d|o)$ versus varying nodding time period for the worst case scenario for the sensor at vehicle velocity, $v = 45 \text{ m/s}$, for a uniform $p(o)$. Each sensor velocity corresponding to different nodding time periods is evaluated till $t_r = 1.4s$. The evaluation shows scans with slower scanning speed are better, this is intuitively the correct behavior as slower nodding speeds means more uniform point distribution of rays in the $[\theta, \phi]$ manifold. Similarly a lower fast axis resolution (Yaw Res.) results in a more uniform distribution of rays in the $[\theta, \phi]$ manifold, leading to more information gain.

Policy for Explicit Data Gathering

The policy for scanning for landing zone evaluation is also computed to maximize the information gain on the LZ. The LZ we are concerned with is defined as a circle of $50m$ radius around a nominal point on the terrain for each mission. Since a fixed region needs to be sampled for LZ evaluation over the length of the approach, the laser is controlled to sample it with a uniform distribution of points, in order to maximize information gain. The sampling rate of the laser is fixed for safety scans to 29000 points per second, as changing the sampling rate means switching off the laser for 7 seconds and reducing the maximum range of the laser from $1400m$ to $650m$, which renders the vehicle unsafe at high speeds. Once the safety of the entire mission is ensured, the sampling rate of the sensor is increased to 83000, if it

enables a denser sampling of LZ.

4.4 Results

Effective Range Increment

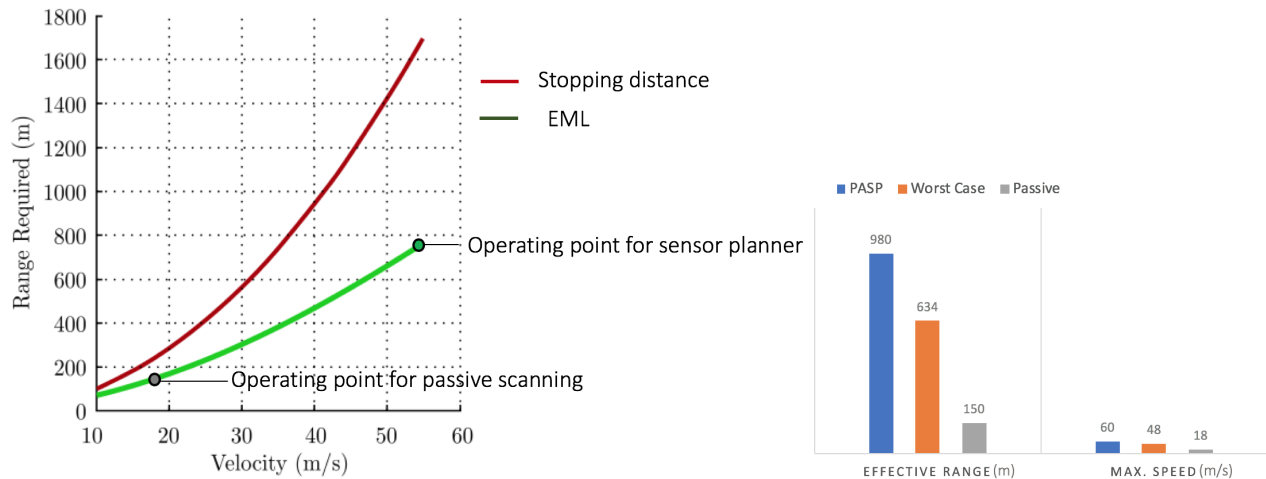


Fig. 4.6 shows the effective range increment of the vehicle by using the sensor planner approach described in this chapter. The effective range of the sensor increases more than six times over passive nodding pattern and the online augmentation of offline worst-case policy led to an effective range increment of roughly 1.5 times. Enabling rotorcraft to fly at speeds of up to 56 m/s safely, while passive scanning approach restricted the maximum safe speed to 18 m/s. We test our approach on multiple full-scale helicopters equipped with Near Earth Autonomy, M3 active nodding range sensor. Over the course of approximately 3 years, the sensor planning and emergency maneuver library based safety system was tested on a number of craft. Tests were facilitated through Boeing, Near Earth Autonomy, Executive Helicopters, and Aurora Flight Sciences. All helicopters shown in table 4.1 had a safety pilot on board. The TALOS system could either execute in a fully autonomous mode where it sent commands directly to the helicopter flight control system or provided such commanded trajectories to the pilot to follow through a nearby display. The Unmanned Little Bird was equipped with fly by wire capability and could switch between fully autonomous and pilot in the loop modes. The Bell helicopters used were not equipped with fly by wire capability, so were only flown with the pilot in command.

As can be seen from table 4.2 many tests of the sensor planning and EML based safety system have been conducted throughout the project. These tests were accomplished with the help of Boeing (Phase I), Executive

Figure 4.6: Left: Effective sensor range increment using the worst-case policy as compared to passive scanning. Right: Effective sensor ranges and max safe speeds for online (PASP), worst-case and passive sensing policies.




Characteristics	Boeing H-6U (Unmanned Little Bird)	Bell 206B (Jet Ranger)	Bell 206L and 206L-3 (Long Ranger)
			
Project phase	Phase I	Phase II and III	Phase III
Engine power	485 kW (650 hp)	310 kW (420 hp)	485 kW (650 hp)
Empty weight	998 kg (2200 lbs)	730 kg (1609 lbs)	998 kg (2200 lbs)
Max takeoff weight	1610 kg (3100 lbs)	1451kg (3200 lbs)	1882 kg (4150 lbs)
Rotor diameter	8.38 m (27.4 ft)	11.28 m (37 ft)	11.28m (37 ft)
Max speed	268 km/h (145 KTAS)	219 km/h (118 KTAS)	232 km/h (125 KTAS)
Rate of climb	10.5 m/s (2070 ft/min)	6.6 m/s (1300 ft/min)	6.6 m/s (1300 ft/min)
Service ceiling	6096 m (20000 ft)	3114 m (13500 ft)	6096 m (20000 ft)

Table 4.1: Helicopter Platforms Comparison

Helicopters (Phase II), Near Earth Autonomy, and Aurora Flight Sciences, who provided both the hardware and staff to perform these tests. Performing tests with a full scale helicopter is a complex task that requires a lot of time and resources for preparation, execution, and post flight evaluation and data analysis. Looking at the table presented above, it is possible to get an average of around 6 test runs per day. Each test run is composed in general by one short flight that takes no more than 15 minutes, from takeoff to land. The low number of test runs per day, reflects the amount of preparation required prior to flight, fixing software or hardware issues, wait time or test cancelled due to weather related events, and unexpected air traffic in the test locations. The test were conducted in Mesa, AZ, Manassas, VA, Pittsburgh, PA and Quantico, VA.

Test Type	Phase I (runs)	Phase I (days)	Phase II (runs)	Phase II (days)	Phase III (runs)	Phase III (days)
Unit Test	(*)	(*)	34	8	214	28
Integrated	24	2	61	11	284	51
Total Test	24	2	95	19	498	79

Table 4.2: Number of test runs performed along the project

Case Study 4.1: A Typical Landing Mission

In this section we describe a typical mission, in which the helicopter performs autonomous landing while being guaranteed safe. The lidar policy switches from scanning for safety to scanning for landing zone evaluation. Fig. 4.7 Shows a typical mission overlaid over satellite map. Fig. 4.8 and Fig. 4.9 show the sensor trajectory and the vehicle trajectory for the same flight. It is interesting to note that during a sharp turn the safe speed provided by emergency maneuver library, only slightly decreases as it better utilizes the vehicle dynamics and free space, whereas the stopping distance based safe velocity drops dramatically. Another interesting artifact of contextual

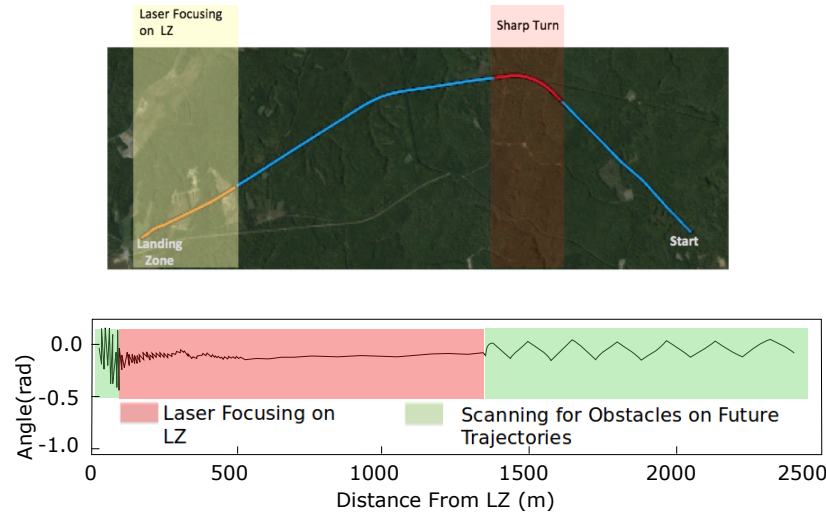


Figure 4.7: A mission at Quantico, 26th Feb 2014

data gathering is visible in Fig. 4.9, where after the laser has scanned the LZ and it starts scanning the region for a possible wave-off or take-off, the maximum safe speed of the vehicle increases from 40 m/s to 60 m/s. With passive scanning or fixed (state-machine based) sensing policies the switch from LZ to gathering information for wave-off will only occur retro-actively, rather than pro-actively.

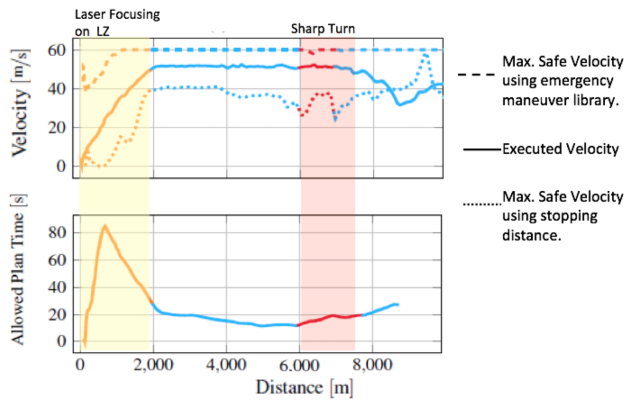
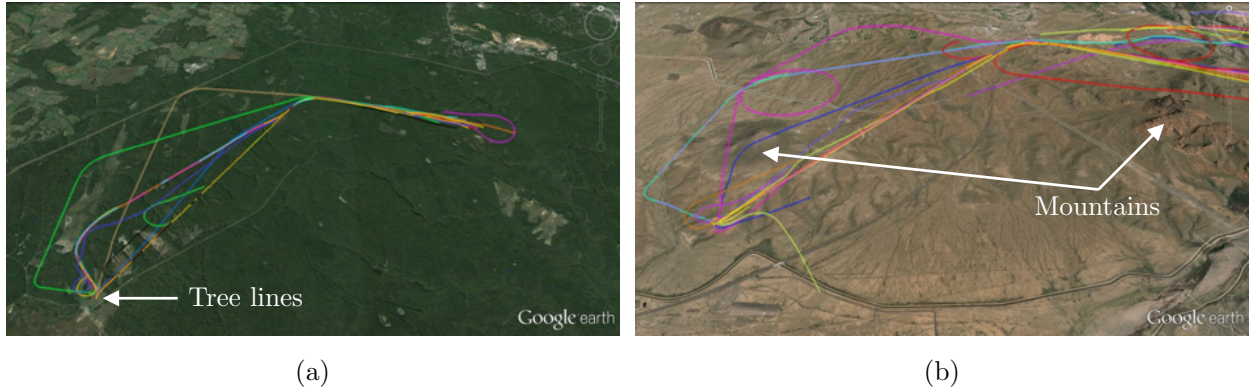


Figure 4.8: Sensor Nodding Profile: The figure shows the sensor’s nodding profile. In the initial part of the mission, the sensor acts to keeps the vehicle safe. It switches to focussing on the landing zone when safety for the remainder of the mission has been guaranteed. This focussing of the laser is shown by the narrow peak to peak of the nodding angles. At the very end, once enough points on the landing zone has been focussed, the sensor reverts back to ensuring that the vehicle can be safe should it desire to waveoff.

Figure 4.9: Safety Benefits: The figure shows the sensor’s nodding profile. In the initial part of the mission, the sensor acts to keeps the vehicle safe. It switches to focussing on the landing zone when safety for the remainder of the mission has been guaranteed. This focussing of the laser is shown by the narrow peak to peak of the nodding angles. At the very end, once enough points on the landing zone has been focussed, the sensor reverts back to ensuring that the vehicle can be safe should it desire to waveoff.

Case Study 4.2: Quantico and Mesa Flights

Fig. 4.10 shows a typical collection of missions on any given flight date and Fig. 4.11 provides the corresponding safety and planning time statistics for the missions. All the missions need the vehicle to navigate from start to the landing zone at high speeds, evaluate the landing zone and land without hovering if the landing zone is safe. The sensor had to clear enough region to keep the vehicle safe, even at the speeds of 56 m/s and sample



a 100m diameter landing zone with enough number of points to evaluate it for landing the vehicle on it. On an average the time available to the laser to sample the landing zone (LZ) was below 20 seconds. Such missions were impossible to conduct with a passive scanning approach. Passive scans designed to ensure safety and evaluate landing zone are only able to keep the vehicle safe up to a maximum speed of 18 m/s and evaluate the complete landing zone in 112 seconds.

Figure 4.10: Mission Definition: The helicopter navigates from loiter point (on the right) to landing zone (on the left), a distance of more than 10km in less than 210 seconds. It has to navigate the environment while being provably safe and touch down at the LZ without hovering over it to look for potential sites. The sensor is controlled to enable safe completion of the mission.

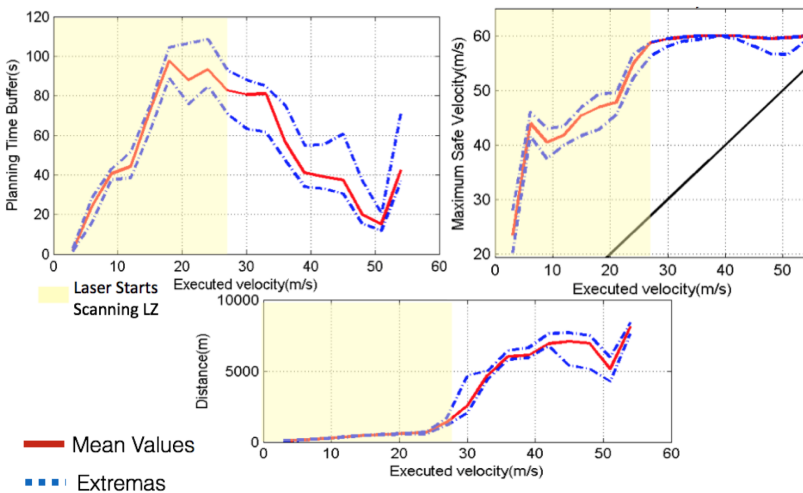


Figure 4.11: Mission Definition: The rotorcraft has to navigate the environment while being provably safe and touch down at the LZ without hovering over it to look for potential sites. The sensor is controlled to enable safe completion of the mission.

4.5 Summary

This chapter defines a policy to control a sensor mounted on the vehicle to maximize the vehicle's performance. We construct and demonstrate a policy function that ensures safety up to a maximum speed if the environment allows so. We proved that the problem of minimizing vehicle trajectory

cost by actively controlling the sensor can also be formulated as that of, optimizing the sensor trajectory to maximize the contextually important expected information gain. We showed that not only the optimization for information gain is NP hard but the evaluation of expected information gain is prohibitively expensive for gradient based optimization. Nevertheless, we have demonstrated a simple policy to control the sensor to maximize contextually important information gain can be used to drastically improve the robot's capabilities. Main contributions of this chapter are -

- *Proving equivalence between minimizing vehicle trajectory and maximizing contextual information gain*- We prove that for a fixed path, optimizing sensor motion for minimizing vehicle trajectory cost is equivalent to maximizing contextual information gain. Using this property the original problem of minimizing vehicle trajectory cost can be converted to a better studied information theoretic approach, Prop. 3.
- *Construction of sensing policy offline to provide performance guarantees*- Constructing worst case for safety, where the sensor has maximize the information gain in the next step, results in a sensor motion policy that can provide offline guarantees on vehicle performance. The worst case policy was learnt offline by exploiting symmetris of constant velocity actions. Enabling a compute effort reduction of the order of 10^{15} , Sec. 4.6.
- *Augmenting worst case policy on-line to improve performance* - The core idea of taking visibility and already sensed regions into account through a sparse set of features enabled a 1.5 times performance increase over worst case sensor motion policy, Fig. 4.6.
- *More than 100kms of autonomous flight testing, over the course last three years*- The sensor planner enabled vehicle to fly safely up to the speeds of 60 m/s and evaluating landing zones in less than 20s, where passive sensory action approaches restricted the maximum speed to just 18 m/s , while taking 112s to evaluate landing zones.

Contributions towards Enabling Safe Data Gathering in Physical Spaces

This work split the decision-theoretic formulation (equation 2.1) of active data gathering in physical spaces in two parts, optimization of sensing motion (equation 4.3) and optimization of vehicle motion. We described an algorithm to optimize sensor motion given a vehicle path. By modeling the dynamic constraints via *EML*, the algorithm provides performance guarantees by optimizing for data gathering in a worst-case scenario for vehicle safety. The offline learnt policy is augmented online to account for visibility and already known regions.

After covering active sensor motion planning for implicit and explicit data gathering, we shift our focus on vehicle trajectory motion planning for

explicit data gathering or Information Path-Planning as it is classically known. In the next chapter we identify an inherent structure in multi-resolution data gathering that renders Markov Decision Process (MDP) based methods ineffective.

4.6 Proofs and Algorithmic Details

Maximizing Information Gain to Minimize Trajectory Cost

Proposition 3 (Maximizing Information Gain to Minimize Trajectory Cost).

Given a vehicle trajectory $\sigma(\cdot)$, optimizing sensing policy to minimize expected cost of the vehicle trajectory, eq. 4.2, is equivalent to optimizing sensor's policy for maximizing contextually important information gain, eq. 4.3.

Proof. Let π_s^* be the the optimal solution to eq. 4.2. Optimizing eq. 4.2 is equivalent to optimizing eq. 4.3, if $\pi_s^*(\cdot)$ is the optimal solution to eq. 4.3 as well. We prove the proposition 3 by constructing $M(\cdot)$ and $\mathbb{IG}(\cdot)$ such that $\pi_s^*(\cdot)$ is trivially the solution of eq. 4.3. Such an $M(\cdot)$ and $\mathbb{IG}(\cdot)$ are defined in equation 4.8 and 4.9.

$$M(x, \sigma(t), b(t)) = \begin{cases} 1 & \text{if } x \text{ was sensed by } \pi_s^*(b(t)), \text{ at } \sigma(t) \text{ and } b(t) \\ 0 & \text{otherwise} \end{cases} \quad (4.8)$$

$$\mathbb{IG}(x, \sigma(t), \pi_s(b(t)), b(t)) = \min(\mathbb{IG}(x, \sigma(t), \pi_s(b(t)), b(t)), \mathbb{IG}(x, \sigma(t), \pi_s^*(b(t)), b(t))) \quad (4.9)$$

By definition, $\pi_s^*(t)$ senses the information at all the locations where $M(x, \sigma(t), b(t)) = 1$, which is its maximum value for all $x \in X$ for any given time t . In order to prove that $\pi_s^*(t)$ is a solution to eq. 4.3, we need to prove that $\pi_s^*(t)$ also maximizes $\mathbb{IG}(\cdot)$. We define $\mathbb{IG}(\cdot)$, such that its maximum possible value for a given x , $\sigma(t)$ and $b(t)$ is obtained by $\pi_s^*(b(t))$. Hence by construction $\pi_s^*(\cdot)$ is a solution to eq. 4.3. It is important to note the obvious fact that solving the optimization problem by maximizing the information gain is as hard as the original problem. ■

Complexity of Calculating Information Gain

Proposition 4 (Complexity of calculating information gain). For a laser sensory action with n_r rays, interacting with N cells of an occupancy grid map, the worst case computation complexity of calculating information gain is $O(3^{n_r} N)$.

Proof. Consider a robot equipped with sensors that provide range measurements to the nearest obstacle. The robot employs occupancy grid map to construct the map and detect obstacles. The occupancy grid models the environment as an n_m -tuple random variable $M = (M^{[1]}, \dots, M^{[n_m]})$, where $M^{[i]}$ is binary random variable that is 0 in absence of obstacle in voxel i and 1 in presence of obstacles. For each voxel i at a given time t , occupancy

grid map stores $l_t^i = \text{logit}(P(M^i = 1|z_{0:t}))$, where $z_{0:t}$ indicate the measurements observed from the ranging lidars. $P_{occ}^{i,t} = P(M^i = 1|z_{0:t})$ or $P_{emp}^{i,t} = P(M^i = 0|z_{0:t})$ can be calculated from the logit function.

Let, R_t^i be the set of all the rays that may pass through a voxel i at time t , given the sensor geometry. Let, $n_{t,r}^i = |R_t^i|$. Let $Z_t^i = [Z_t^{i,1}, \dots, Z_t^{i,n_{t,r}^i}]$ be a $n_{t,r}^i$ -tuple random variable, where $Z_t^{i,j}$ is an independent ternary random variable that is z_{occ} if the ray j observes an obstacle in voxel i at time t , z_{emp} if voxel i is empty or z_ϕ if ray j does not interact with voxel i . Let, Γ_t^i be the set of all possible values of random variable Z_t^i . The number of possible values of Z_t^i or the cardinality of Γ_t^i is given by $3^{n_{t,r}^i}$. Let, $\gamma \in \Gamma_t^i$ be defined as $\gamma = [z^1, \dots, z^{n_{t,r}^i}]$, where, $z^j \in z_{occ}, z_{emp}, z_\phi$. Then equation 4.10 gives the probability of $\gamma, P(\gamma)$.

$$P(\gamma) = \prod_{j=1}^{n_{t,r}^i} P(z^j) \quad (4.10)$$

where, $P(z^j)$ can be calculated using algorithm 6.

Let, I^j be an ordered set of voxel indices through which a ray $j \in R_t$ passes through. Let, $I^j(v)$ be a function, returns the index of the v^{th} voxel that ray j traverses. $P_j^i(z)$, $z \in z_{occ}, z_{emp}, z_\phi$ stores the probability of ray j observing either z_{occ} , z_{emp} or z_ϕ in voxel i . If we know that a ray interacts with a voxel, the probability of observing that voxel is occupied given its state is $P(z_{occ}^{[i]}|M^{[i]})$ and is constant for all the $M^{[i]} \in M$. Also, $P(z_{emp}|M^{[i]}) = 1 - P(z_{occ}|M^{[i]})$. Note that any ray casting method can be used to generate I^j , R_t^i for all rays $j \in R_t$ and cells i in M .

The expected information gain at a time t , for a voxel i , from which n_r rays pass, is then given by equation 4.11

$$\mathbf{IG}_{n_r}^i = H(M^{[i]}|Z_{0:t-1}) - \sum_{\forall \gamma \in \Gamma_t^i} \prod_{\forall z^j \in \gamma} P(z^j) H(M^{[i]}|\gamma, Z_{0:t-1}) \quad (4.11)$$

where, $H(\cdot)$ is the Shannon entropy [Shannon, 1951], both $M^{[i]}|Z_{0:t-1}$ and $M^{[i]}|\gamma, Z_{0:t-1}$ can be calculated through equation 4.12.

$$l_t^i = l_{t-1}^i + \sum_{\forall j \in Z_t^i} \text{logit}(P(M^i = 1|j)) \quad (4.12)$$

As discussed before if n_r rays pass through a cell, there are 3^{n_r} possible unique combinations of measurements for that cell. If there are N voxels through which maximum n_r rays pass. The computation complexity of evaluating expected information gain is given by $O(3^{n_r} N)$. ■

Calculating Features

Algorithm 7 describes the process to compute the features (ζ_m, r_e) , as mentioned in section 4.2. ζ_m is a 2 dimensional map along pitch and yaw axis. A

Algorithm 6: MeasurementProbability()

Require: $M, I_j, P(z_{occ}^{[i]}|M^{[i]}), P(z_{emp}^{[i]}|M^{[i]}) \forall i \in M$

for $v = 1 : |I_j|$ **do**

if $v = 1$ **then**

$P_j^{I(v)}(z_{occ}) \leftarrow P(z_{occ}^{[I(v)]}|M^{I(v)} = 0)P(M^{I(v)} = 0) + P(z_{occ}^{[I(v)]}|M^{I(v)} = 1)P(M^{I(v)} = 1);$
 $P_j^{I(v)}(z_{emp}) \leftarrow P(z_{emp}^{[I(v)]}|M^{I(v)} = 0)P(M^{I(v)} = 0) + P(z_{emp}^{[I(v)]}|M^{I(v)} = 1)P(M^{I(v)} = 1);$
 $P_j^{I(v)}(z_\phi) \leftarrow 0;$

end

else

$P_j^{I(v)}(z_{occ}) \leftarrow P_j^{I(v-1)}(z_{emp})(P(z_{occ}^{[I(v)]}|M^{I(v)} = 0)P(M^{I(v)} = 0) + P(z_{occ}^{[I(v)]}|M^{I(v)} = 1)P(M^{I(v)} = 1));$
 $P_j^{I(v)}(z_{emp}) \leftarrow P_j^{I(v-1)}(P(z_{emp}^{[I(v)]}|M^{I(v)} = 0)P(M^{I(v)} = 0) + P(z_{emp}^{[I(v)]}|M^{I(v)} = 1)P(M^{I(v)} = 1));$
 $P_j^{I(v)}(z_\phi) \leftarrow P_j^{I(v-1)}(z_\phi) + P_j^{I(v-1)}(z_{occ});$

end

end

$P_j^i(z_\phi) \leftarrow 1 \forall i \notin I_j;$

Return : $P_j^i(z) \forall z \in z_{occ}, z_{emp}, z_\phi, \forall i \in M$

given cell of the map stores contextually weighted expected information gain along (θ_i, ϕ_j) where θ_i is the pitch and ϕ_j is the yaw at the cell $[i, j]$. For each cell in ζ_m , we cast a ray along the corresponding pitch and yaw direction. The function $RayCast(\theta_i, \phi_j)$ returns the indices of all the occupancy grid voxels that a ray along (θ_i, ϕ_j) might visit. $MeasurementProbability(\cdot)$ function as described in algorithm 6 returns the probability of observing each cell along the ray as either occupied or empty. Eq. 4.11 defines the expected information gain for a given voxel and eq. 4.13 provides the definition of contextual importance. The inputs to the feature calculation algorithm is the observation model of the laser, occupancy grid map.

For calculating the features we want to capture whether a ray might sense any information that is contextually important to the mission at any given time. We define $CI(x)$, as the contextual importance of the volume occupied by the voxel at point x .

$$CI(x) = 1, if M(x, \sigma(k), b(t)) > 0 \text{ for any } k \in [t, T] \quad (4.13)$$

where, T is the final time of the mission.

Section 4.2 describes how ζ_m is used to calculate the relevant field of view for the sensor to scan. In order to make an informed decision about the scanning action we need to know at what range the information is required. We compute r_e as the average range at which contextually weighted importance is received. To make sure the range at which information is requested is sufficient to ensure safety, it is lower bounded by the range required to make the vehicle safe. $R_{safe} : \mathcal{X} \rightarrow \mathbb{R}$ is the function that returns the minimum range required to keep a state safe. Given an emergency maneuver library this function is trivial to compute.

Worst Case Scenario for Safety

Proposition 5 (Worst Case Scenario for Safety). *The sensor would have to reduce uncertainty for maximum volume in the environment if the vehicle is moving at the maximum speed that allows it to stay safe at r_e .*

Proof. Let the known volume be V_k . To guarantee safety of the vehicle if the environment allows so, the sensor has to reduce uncertainty about the presence of obstacle of interest in $V_s(\sigma(t + t_r)) - V_k$. For the worst case scenario the volume in which information is to be gained or uncertainty to be reduced is maximum. Therefore for the worst case scenario we have to maximize $V_s(\sigma(t + t_r))$. V_k is fixed, so we have to maximize $V_s(\cdot)$.

$V_s(\sigma(t + t_r))$ is monotonic in speed at $\sigma(t + t_r)$. Hence, we want to maximize the speed at $\sigma(t + t_r)$. Which implies the sensor will have to scan maximum volume in minimum time if the vehicle is moving at maximum safe speed possible for r_e . ■

Obviously, if the sensor can ensure safety in the worst case scenario for all

Algorithm 7: FeatureCalculator()

Require: $M, P(z_{occ}|M = 0), P(z_{occ}|M = 1)$
 $voxelcount = 0;$
for $\theta = \theta_{min} : \theta_{res} : \theta_{max}$ **do**
 for $\phi = \phi_{min} : \phi_{res} : \phi_{max}$ **do**
 $I^{\theta,\phi} = RayCast(\theta, \phi);$
 $P_{meas}(\cdot) = MeasurementProbability(M, I^{\theta,\phi}, P(z_{occ}|M = 0), P(z_{occ}|M = 1));$
 $\mathbb{I}G_{\theta,\phi} = 0;$
 $r_e = 0;$
 for $i \in I^{\theta,\phi}$ **do**
 $\mathbb{I}G_{\theta,\phi} = CI(i_x) \mathbb{I}G^i + \mathbb{I}G_{\theta,\phi};$
 // i_x is the position of the center of the voxel i in the workspace;
 $r_e = r_e + CI(i_x) \mathbb{I}G^i Range(i);$
 $voxelcount ++;$
 end
 $\zeta_m(\theta, \phi) = \mathbb{I}G_{\theta,\phi};$
 end
end
 $r_e = max(r_e/voxelcount, R_{safe}(\sigma(t + t_r)));$
Return : ζ_m, r_e

r_e , then the vehicle can fly at its maximum speed safely if the environment allows so.

Monotonicity of Expected Information Gain

Proposition 6 (Monotonicity of Expected Information Gain). *Assuming there are no false positives, $p(d|o') = 0, p(d'|o') = 1$. For a given $p(o)$, $\mathbb{I}G$ can be completely expressed as a function of $p(d|o)$ and monotonically increases in $p(d|o)$.*

Proof. Assuming there are no false positives, $p(d|o') = 0, p(d'|o') = 1$. For brevity we change the notation to $p(d|o) = p_a, p(d'|o) = p'_a$. The equation reduces to the following-

$$\begin{aligned} \mathbb{I}G(o) = & p(o)(p(d|o)[H(p(o)) - H(p(o|d))] + \\ & p(d'|o)(H(p(o)) - H(p(o|d')))] + \\ & p(o')(p(d|o')(H(p(o)) - H(p(o|d))) + \\ & p(d'|o')(H(p(o)) - H(p(o|d')))) \end{aligned} \quad (4.14)$$

where, $H(\cdot)$ is the entropy, [Shannon, 1951].

$$\begin{aligned} \mathbb{I}G(o) = & H(o) - p(o)p_a H(o|d) - \\ & p(o)p'_a H(o|d') - p(o')H(o|d') \end{aligned} \quad (4.15)$$

Given $p(d|o') = 0, p(d'|o') = 1, p(o|d) = 1$, can be easily verified using Baye's Rule. Since $p(o|d) = 1, H(o|d) = 0$.

$$\mathbb{I}G(o) = H(o) - (1 - p(o)p_a)H(o|d') \quad (4.16)$$

where,

$$H(o|d') = - \sum_o p(o|d') \log(p(o|d')) \quad (4.17)$$

$p(o|d')$ can be expressed as a function of p_a and $p(o)$, and is given by following equation.

$$p(o|d') = (1 - p_a)p(o) / (1 - p_a p(o)) \quad (4.18)$$

Using equation 4.16, 4.17 and 4.18 the expected information gain can be completely expressed as a function of $[p(d|o) = p_a, p(o)]$. Obviously expected information gain is zero if $p(d|o) = 0$ and positive otherwise, monotonically increasing with $p(d|o)$. ■

Probability of Detection vs. Distance

Proposition 7 (Probability of Detection vs. Distance). *Given a fixed angular motion profile of a lidar sensor, if an object is detected when n laser hits*

are reported on it, and the probability of an intersecting ray reporting a hit is p_h . The probability of detection of the object decreases with increase in the distance between the object and the lidar sensor, assuming the relative orientation of the object and lidar is constant.

Proof. Given a fixed angular motion of the laser the number of rays that intersect a given object decreases with increasing distance between the object and laser. The decrease in number of hits results in decrease in the probability of detection of the object. Assuming an object is placed at a distance such that N_r rays intersect with the object. We show that probability of detection of an object is greater when $N_r + 1$ rays intersect than when N_r rays intersect it. The probability on detecting an object when N_r rays intersect ($P_d^{N_r}$), it is given by equation 4.19.

$$P_d^{N_r} = \sum_{k=n}^{N_r} \binom{N_r}{k} p_h^k (1 - p_h)^{N_r - k} \quad (4.19)$$

If $N_r + 1$ rays intersect the object the probability of intersection is given by equation 4.20.

$$\begin{aligned} P_d^{N_r+1} &= \sum_{k=n}^{N_r} \binom{N_r}{k} p_h^k (1 - p_h)^{N_r - k} + \binom{N_r}{n-1} p_h^{n-1} (1 - p_h)^{N_r - n + 1} P_d^{N_r+1} \\ &= P_d^{N_r} + \binom{N_r}{n-1} p_h^{n-1} (1 - p_h)^{N_r - n + 1} \end{aligned} \quad (4.20)$$

Since, $\binom{N_r}{n-1} p_h^{n-1} (1 - p_h)^{N_r - n + 1} \geq 0$.

$$P_d^{N_r+1} \geq P_d^{N_r} \quad (4.21)$$

■

Reduction in Computational Complexity for Evaluating Actions

The algorithm described in section 4.3, calculates $P(d|o)$ by first projecting the object at given distance r and orientation α, β, γ in the $[\phi, \theta]$ plane of spherical co-ordinates at a nominal range of $1m$. The laser rays are also projected to the same plane. Both the object and rays can be represented as polygons in the $[\phi, \theta]$ plane. The rays are then projected to the configuration space of the object. $P(d|o)$ is then easily calculable for all ϕ, θ at r , through calculating ray polygonal intersections in c-space. The polygonal intersections can be calculated in $O(R^3)$ in the worst case. Since, the $P(d|o)$ is calculated for all ϕ, θ , it eliminates the need to sample in those dimensions, leading to reduction in number of particles to N^4 . The computational complexity of calculating $P(d|o)$ is then given by $O(R^3 N^4)$.

Since we are sensing to ensure vehicle safety, we maximize the information gain at the furthest range we are interested in. This reduces the

evaluation of $P(d|o)$ to a ϕ, θ manifold at the range r_e . Which eliminates the need to sample in dimension r . Since, we are interested in wires of length l , aligned with the ϕ, θ manifold, the sampling is only restricted to the orientation of the object in the ϕ, θ manifold. The computation complexity as a result reduces to $O(R^3N)$.

Further, we reduce the action space to constant velocity nods, which results in a grid of laser ray polygons in the $[\phi, \theta]$ manifold. It is obvious that for an infinite grid the $P(d|o)$ function will repeat itself for each cell. Hence, if we make an infinite grid assumption, $P(d|o)$ need only be evaluated for all the configurations of the object within a cell. If the resolution of the grid generated by laser ray polygons in the $[\phi, \theta]$ manifold is given by $[\phi_{res}, \theta_{res}]$. The number of polygons that may interact with the object to be detected is given by $R_l = \max(l + \phi_{res}, l + \theta_{res})^2 / \min(\theta_{res}, \phi_{res})^2$. Hence, the computational complexity reduces to $O(R_l^3N)$. In a nod of 1.4s for the helicopter at moving at 45m/s, the length of wire projected in the $[\phi, \theta]$ manifold is, $l = 1.2^\circ$, $\phi_{res} = 0.48^\circ$, $\theta_{res} = 0.54^\circ$, $R_l = 11$. We use $N = 10000$ samples for each dimension. Therefore the computational cost reduction is of the order of 10^{15} .

Modeling of Belief Space Dynamics for Budgeted, Multi-Resolution Data Gathering

In chapter 3 we described the safety constraint and an algorithm to enforce it online. In chapter 4 we described the sensor motion optimization for active data gathering in physical spaces that leverage the safety definition to model dynamic constraints of the vehicle and enable data gathering for safety. In this chapter, we shift our focus on vehicle motion planning for active data gathering for task specific requirements. We characterize the decision-theoretic formulation of motion planning for active data gathering and show that for agents with multi-resolution data gathering capabilities it is imperative to model high-dimensional belief spaces dynamics to generate efficient solutions. We start by briefly describing Markov Decision Processes and Partially Observable Markov Decision Processes as decision-theoretic solvers.

Markov Decision Processes (MDPs) were developed as a part of stochastic control theory [Beutler, 1989]. The MDP formulation of problems makes underlying assumptions that the state of the environment and the agents is known, while dynamics can be stochastic with known parameters and the state is Markovian, [LaValle, 2006]. MDPs are computationally relatively efficient to solve [Littman et al., 1995b]. Therefore, in robotics applications, whenever it is reasonable to make the assumption that the world is completely known or the uncertainty in the world can be ignored, MDP has been a useful tool to solve sequential decision making problems [Choudhury et al., 2015, Bakker et al., 2005, Trautman and Krause, 2010, Pivtoraiko and Kelly, 2011, Abbeel et al., 2007]. However, for the problems where uncertainty cannot be ignored POMDPs were developed [Kaelbling et al., 1998].

POMDPs are computationally intractable to solve optimally in the worst case [Papadimitriou and Tsitsiklis, 1987]. Approximate POMDP solvers have been developed to solve POMDPs tractably, we refer readers to [Shani et al., 2013, Ross et al., 2008] for a comprehensive literature review. A popular approach to solving POMDPs approximately is to leverage MDP solvers and take the best expected action assuming the uncertainty about

the environment will not change or it will magically disappear in the next step, FF-Re-plan [Little et al., 2007], Hindsight optimization [Yoon et al., 2008] and QMDP [Littman et al., 1995a] are examples of such methods. Despite strong assumptions these methods found early success as leading online POMDP solvers and have been effectively applied to problems like shared autonomy [Javdani et al., 2015], manipulation planning [Koval et al., 2016b,a] etc. We call such methods MDP based approximate POMDP solvers, MDP-POMDP solvers in short.

Use of MDP based approximate POMDP solvers requires computing expectation over either prior or posterior of the distribution over the state space. For large state spaces computing this expectation online can be challenging, to overcome this challenge, imitation learning techniques are used in conjunction with MDP solvers to train a policy offline such that it implicitly takes the expectation over uncertainty over state space and picks the best action online [Tamar et al., 2017, Choudhury et al., 2017a, Zhang et al., 2016].

MDP based approximate POMDP solvers, including the imitation learning based methods that rely on MDPs fail at taking actions that do not belong to optimal MDP policies. It is well known, actions that lead to reduction in uncertainty about the state space, while not providing rewards are an important class of actions that are often not included in optimal MDP policies [Yoon et al., 2008, Littman et al., 1995a, Choudhury et al., 2017b]. The failure of agents to take such actions can lead to sub-optimal POMDP solutions, sometimes, catastrophically so. *We provide a formal definition for such actions in this chapter and identify with examples the conditions under which MDP based approximate POMDP solvers fail. The presence of actions identified in this chapter, can be confirmed without solving full POMDP problems, and for any problem where such actions exist, the readers are advised to exercise caution against using MDP-POMDP solvers. We also show that such actions do exist for multi-resolution informative path planning problems.* The primary need to identify problems where MDP-POMDP solvers do not work is to avoid the implementation effort before realizing the unsuitability and to make better solver design choices.

Consider the famous tiger problem, [Kaelbling et al., 1998], there is a tiger hidden behind one of the two doors in front of the agent, the agent has three possible actions. 1. Listen to determine the door behind which the tiger is hidden. 2. Open the door without the tiger for high reward 3. Open the door with the tiger for a high penalty. When the uncertainty about tiger's location is removed the optimal MDP policy is to open the door without the tiger. The action to listen for the location of tiger is never a part of the optimal MDP policy. Hence, if there is a 50% chance that the tiger is behind left door and the agent is using MDP-POMDP solvers, it will never listen for location of the tiger and have only a 50/50 chance of opening the door with a tiger in it. Whereas, if the agent would have listened for the the presence of the tiger,

it could have been guaranteed to avoid the tiger. Knowing the tiger problem is unsuitable for MDP-POMDP solvers, we can avoid those methods while designing an approach to solve the problem.

We formalize the conditions under which MDP-POMDP based solvers fail alongside the actions symptomatic of such conditions in section 5.3. We then provide a detailed example of the tiger problem and multi-resolution, budgeted informative path planning problem where such conditions exist and MDP-POMDP solvers can lead to unacceptably poor performance in section 5.4. Before going further we formally re-introduce the MDP and POMDP formulations in section 5.1 for the sake of completeness and briefly describe how MDP solvers are used solve POMDP problems approximately in section 5.2.

5.1 Explicit Data Gathering

MDP Formulation

A markov decision process can be represented by the tuple $MDP = (\mathcal{S}, s_0, \mathcal{A}, \Omega, R, Term)$ where

- \mathcal{S} is a set of states.
- s_0 is the state at time 0, initial state.
- \mathcal{A} is a set of actions.
- Ω is a set of state transition probabilities.
- $R : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$ is the reward function.
- $Term : \mathcal{S} \times \mathcal{A} \rightarrow \{0, 1\}$ is the terminating condition.

At each time step, the agent takes an action $a \in \mathcal{A}$ which causes the environment to transition from state s to state $s' \in \mathcal{S}$ with probability $\Omega(s, a, s') = P(s_{t+1} = s' | s_t = s, a_t = a)$. The agent receives a reward $R(s, a)$. On reaching the new state s' . The MDP is terminated if $Term(s, a) = 1$.

$\pi_M : \mathcal{S} \rightarrow \mathcal{A}$ be a policy function for the MDP that maps from a state to action. Let the state distribution induced by a policy π_M after t time steps, starting with state s_0 be $P(s | \pi_M, s_0, t)$. The value of a policy π_M is the expected cumulative reward for executing π_M for T time steps on the induced state and history distribution

$$J(\pi_M) = \sum_{t=1}^T \mathbb{E}_{s_t \sim P(s_t | \pi_M, s_0, t)} [R(s_t, \pi_M(s_t))] \quad (5.1)$$

The optimal policy for the MDP maximizes the expected cumulative reward, i.e $\pi_M^* \in \arg \max_{\pi_M \in \Pi_M} J(\pi_M)$.

The value of a state s for a given policy π_M is given by $V^{\pi_M}(s) = \sum_{i=1}^T \mathbb{E}_{s_i \sim P(s_i | \pi_M, s)} [R(s_i, \pi_M(s_i))]$

An MDP solver takes an MDP as an input and returns the optimal policy π_M^* .

POMDP Formulation

In most of the real world situations the state of the environment cannot be fully known. Agents rely on observations to infer the state of the environment. Since, the state of the environment is inferred, the agent maintains a probability distribution over possible set of states that the environment can be in. Let, that distribution be $b : \mathcal{S} \rightarrow [0, 1]$. At each time step, the environment is in some state $s \in \mathcal{S}$ which cannot be directly observed. Let, the initial belief be given by $b_0(\cdot)$. The agent takes an action $a \in \mathcal{A}$ which causes the environment to transition to state $s' \in \mathcal{S}$ with probability $\Omega(s, a, s')$. The agent receives a reward $R(s, a)$. On reaching the new state s' , it receives an observation $o \in \mathcal{O}$ according to the probability $Z(s', a, o) = P(o_{t+1} = o | s_{t+1} = s', a_t = a)$. $b_t(\cdot)$ is the state of the belief of the agent at time then $b_{t+1}(\cdot)$, given an action a_t and observation o_{t+1} by

$$b_{t+1}(s') = \eta Z(s', a, o) \sum_{s \in \mathcal{S}} \Omega(s, a, s') b(s) \quad (5.2)$$

where η is a normalization constant.

$\pi(b_t) : B \rightarrow \mathcal{A} \in \Pi$ be a policy function for the MDP that maps from a belief state to action.

The reward of a belief state is given by

$$R_B(b_t, a_t) = \sum_{s \in \mathcal{S}} R(s, a_t) b(s) \quad (5.3)$$

The value of a policy π is the expected cumulative reward for executing π for T timesteps on the induced belief distribution. Given a starting belief b , let $P(b' | b, \pi, i)$ be the induced belief distribution after i timesteps.

$$J(\pi) = \sum_{t=1}^T \mathbb{E}_{b_t \sim P(b_t | \pi, t)} [R_B(b_t, \pi(b_t))] \quad (5.4)$$

The optimal policy maximizes the expected cumulative reward, i.e $\pi^* \in \arg \max_{\pi \in \Pi} J(\pi)$.

The value of executing a policy π from a belief b is the expected cumulative reward:

$$\tilde{V}^\pi(b) = \sum_{i=1}^T \mathbb{E}_{b_i \sim P(b_i | b, \pi, i)} [R_B(b_i, \pi(b_i))] \quad (5.5)$$

The Q-value function $\tilde{Q}^\pi(b, a)$ is defined as the expected sum of one-step-

reward and value-to-go:

$$\begin{aligned} \tilde{Q}^\pi(b, a) = & R_B(b, a) + \\ & \mathbb{E}_{b' \sim P(b'|b, a)} [\tilde{V}^\pi(b')] \end{aligned} \quad (5.6)$$

POMDP provides an elegant framework to formalize and tackle planning problems for agents operating in partially known environments. There are two major challenges that POMDP solvers face: (1) *Keeping track of evolving uncertainty about the state space over the planning horizon.* As future observations need to be accounted for, the solver needs to keep track of future beliefs that are exponential with respect to number of future observation steps. (2) *Computing the expectation over the state space.* Since the state space of most of the problems worth solving is large, computing an expectation over such state space needs large computation, making it expensive to evaluate online. In the next section we explore how MDP-POMDP solvers overcome these challenges.

5.2 Solving POMDP through MDP Solvers

Approximate POMDP solvers like Hindsight Optimization [Yoon et al., 2008] and QMDP [Littman et al., 1995a] leverage MDP solvers and simplifying assumptions about the environment uncertainty to provide approximate solutions to POMDP. Hindsight optimization finds the optimal action assuming that uncertainty about the state cannot be changed, whereas QMDP assumes that the uncertainty will magically disappear after the next action. For both these approaches, action for a given belief b is given by equation 5.17. The only difference is that the value function computed by using QMDP is a tighter upper-bound than Hindsight optimization for the POMDP value function.

$$a = \arg \max_{a \in \mathcal{A}} \mathbb{E}_{s \sim b(s)} \left[\max_{\pi_M \in \Pi_M} Q^{\pi_M}(s, a) \right] \quad (5.7)$$

Through the simplifying assumptions about the evolution of belief state, these approaches overcome the first challenge for POMDP solvers — keeping track of evolving uncertainty over planning horizon. However, these approaches still need the expectation over the state space to be computed.

Imitation learning based approaches [Tamar et al., 2017, Choudhury et al., 2017a, Zhang et al., 2016] address this concern through data driven techniques. MDP solvers are used over sampled MDP problems to train a policy on the expected distribution of problems. Enabling the online policy to take the best decision in expectation if magically all the uncertainty would disappear in the next step.

Family of approximate POMDP solvers that use MDP work quite well in problems where the required changes in belief can be attained by actions that are rewarding as well. A well-known problem with MDP solvers is that they

do not take actions that don't belong to optimal MDP policies. Hence, neither do the imitation learning methods.

A class of such actions is informative actions. MDP solvers have no motivation to gain information, since under their assumption, either the environment is already known as much as it can be or the uncertainty will disappear after the next step but for POMDP problems gaining information can be useful, [Littman et al., 1995a].

In the next section we show that gaining information through observations can only lead to a belief state with higher value, and discuss a class of information gaining actions that will not be taken by an MDP solver leading to sub-optimal solutions by MDP-POMDP solvers.

5.3 Expected Value of Information

Information is gained through observing the environment. Let a POMDP be given by $POMDP = (S, b_0, \mathcal{A}, \Omega, R, Z, Term)$. Let $o \in \mathcal{O}$ be observed and the belief changes from b to b^o . The Q -value of the b^o is given by equation 5.8.

$$\begin{aligned} \tilde{V}^{\pi^*}(b^o) = \tilde{Q}^{\pi^*}(b^o, a) = \max_{a \in \mathcal{A}} & \left[R_B(b^o, a) + \right. \\ & \left. \mathbb{E}_{b' \sim P(b'|b^o, a)} [\tilde{V}^{\pi^*}(b')] \right] \end{aligned} \quad (5.8)$$

Let's say the probability of observing observation o given a belief b is $P(o|b)$.

Definition 2: *Expected value of information is given by the difference between the expected of value of belief state reached after making an observation and the optimal value of the belief state before the observation, equation 5.9.*

$$\mathcal{EVI}_o(b) = \mathbb{E}_{o \sim P(o|b)} \left[\max_{a \in \mathcal{A}} \tilde{Q}^{\pi^*}(b^o, a) \right] - \tilde{V}^{\pi^*}(b) \quad (5.9)$$

Theorem 1. *Expected value of information is greater than or equal to 0, for all observation and belief.*

Proof. If we ignore the observation and take the same action that maximizes the Q -value of b , instead of b^o , then the observation does not affect the actions, and the expected value of the belief state. Lets say, the optimal action for b is given by $a_b^* = \arg \max_{a \in \mathcal{A}} \tilde{Q}^{\pi^*}(b, a)$. We replace $\max_{a \in \mathcal{A}}$ with a_b^* in equation 5.9, refer to equation 5.10.

$$\mathbb{E}_{o \sim P(o|b)} \left[\tilde{Q}^{\pi^*}(b^o, a_b^*) \right] - \tilde{V}^{\pi^*}(b) \quad (5.10)$$

Expanding equation 5.10.

$$\sum_{o \in \mathcal{O}} \sum_{b' \in B} \tilde{V}^{\pi^*}(b') P(b'|b, o, a_b^*) P(o|b) - \tilde{V}^{\pi^*}(b) = 0 \quad (5.11)$$

Marginalizing over o .

$$\begin{aligned} & \sum_{s \in \mathcal{S}} R(s, a_b^*) P(s|b) + \\ & \sum_{b' \in B} \tilde{V}^{\pi^*}(b') P(b'|b, a_b^*) - \tilde{V}^{\pi^*}(b) \quad (5.12) \\ & \tilde{V}^{\pi^*}(b) - \tilde{V}^{\pi^*}(b) = 0 \end{aligned}$$

Since we replaced a max function with a fixed action, we can infer that the minimum value of $\mathcal{E}\mathcal{V}\mathcal{I}_o(b)$ is 0. \blacksquare

Unfortunately, optimal MDP solvers do not take informative actions if the actions do not provide with reward or access to more rewarding states.

Definition 3: *Informative actions (a_I) are actions that are not a part of the optimal MDP policy for any state or time, but lead to observations. Formally, they are defined by following set of conditions:*

- $a_I \neq \pi^*(s) \forall s \in \mathcal{S}$.
- $P(b'|b, a, o) = P(b'|b, o)P(o|b)$, where $P(b^o|b, o) = 1$ and $P(b'|b, o) = 0 \forall b' \neq b^o$.

If informative actions lead to more valuable information that they cost, then MDP-POMDP solvers are provably sub-optimal.

Theorem 2. *If there exists an informative action $a_I \in \mathcal{A}$ in POMDP, such that the expected value of information attained by a_I is $\geq -R_b(b, a_I)$, then family of MDP-POMDP solvers will be sub-optimal by atleast $\mathcal{E}\mathcal{V}\mathcal{I}_o(b) + R_B(b, a_I)$.*

Proof. The difference between the Q-value of a_I and Q-value of a_b^* is given by equation 5.13.

$$\tilde{Q}^{\pi^*}(b, a_I) - \tilde{Q}^{\pi^*}(b, a_b^*) \quad (5.13)$$

$$= R_B(b, a_I) + \mathbb{E}_{b' \sim P(b'|b, a)} \left[\tilde{V}^{\pi^*}(b') \right] - \tilde{V}^{\pi^*}(b) \quad (5.14)$$

Since an observation is observed immediately after taking action a_I and definition of a_I , equation 5.14 reduces to equation 5.16.

$$R_B(b, a_I) + \mathbb{E}_{o \sim P(o|b)} \left[\max_{a \in \mathcal{A}} \tilde{Q}^{\pi^*}(b^o, a) \right] - \tilde{V}^{\pi^*}(b) \quad (5.15)$$

$$R_B(b, a_I) + \mathcal{E}\mathcal{V}\mathcal{I}_o(b) \quad (5.16)$$

■

In the next section we show two examples, where MDP-POMDP solvers catastrophically fail on account of their inability to take informative actions, Multi-resolution informative path planning being one of them. We first start with a detailed tiger problem.

5.4 Examples

Tiger Problem

In the canonical tiger problem there are two doors and a tiger is hidden behind one of them. There are two possible states, either the tiger is hidden behind the left door \mathcal{T}_L or the right door \mathcal{T}_R , $\mathcal{S} = \{\mathcal{T}_L, \mathcal{T}_R\}$. The action set is given by, $\mathcal{A} = \{d_L, d_R, l\}$, where d_L denotes opening the door at left, d_R denotes opening the door at right and l denotes the action of listening to determine the location of the tiger. Action l can return observations, $\mathcal{O} = \{o_L, o_R\}$, where o_R denotes a tiger is heard behind the right door and o_L denotes a tiger is heard behind the left door. Let the probability of hearing a tiger behind a door $X \in L, R$ if there is a tiger behind door X is given by $P(o_X | \mathcal{T}_X) = 1$.

The reward of the action state pairs is given by table 5.4.

	$s = \mathcal{T}_L$	$s = \mathcal{T}_R$
$a = d_L$	$R(\mathcal{T}_L, d_L) = 0$	$R(\mathcal{T}_R, d_L) = 100$
$a = d_R$	$R(\mathcal{T}_L, d_R) = 100$	$R(\mathcal{T}_R, d_R) = 0$
$a = l$	$R(\mathcal{T}_L, l) = -1$	$R(\mathcal{T}_R, l) = -1$

Let the initial belief be $b_0(\mathcal{T}_L) = P(\mathcal{T}_L) = 1 - P(\mathcal{T}_R) = 0.5$. If the uncertainty is removed from this problem, the optimal MDP solver will suggest the action to open the door with no tiger behind it, $a = d_L$ or $a = d_R$. And since the action l is neither rewarding or does not lead to a rewarding state, it will never be taken by the MDP solver. Approaches like hindsight optimization and Q -MDP use MDP solvers to compute the next action. Both these approaches find the optimal action assuming that uncertainty about the state cannot be changed. The action is given by equation 5.17.

$$a = \arg \max_{a \in \mathcal{A}} \mathbb{E}_{s \sim b(s)} \left[\max_{\pi_M \in \Pi_M} Q^{\pi_M}(s, a) \right] \quad (5.17)$$

For the tiger example with $b_0 = 0.5$, equation returns actions \mathcal{T}_L or \mathcal{T}_R with the hindsight upper-bound value $\tilde{V}_{hs}^{\pi_M^*}(b_0) = 100$, whereas the actual value of action \mathcal{T}_L or \mathcal{T}_R is 50. However, if the action l is taken, the state of

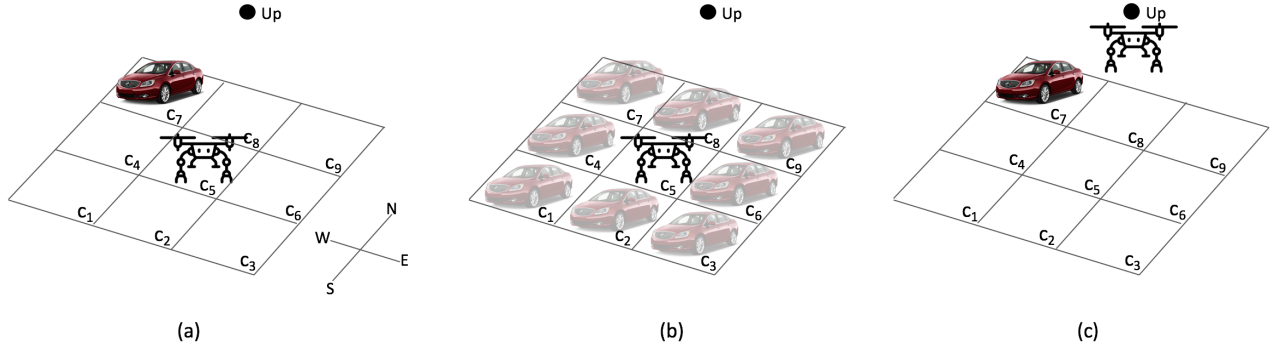


Figure 5.1: (a) The UAV gets a high reward if it visits the same cell as the car to gather high resolution information of the car (lets say its numberplate.). Since the UAV knows where the car is, it does not need to take action up and can directly go to the cell with the car. (b) In the POMDP version of the problem the location of the car is unknown, hence there is only a small chance that the next cell the UAV goes to has a car in it, leading to a small expected reward. (c) But if UAV gains height, the uncertainty about the location of the car is removed. Although there is no reward from this action but the removal uncertainty leads to guarantee of gaining high reward by visiting the cell in which the car exists. Since, MDP-POMDP solvers cannot take informative action up , they are sub-optimal for such data gathering problems.

the environment is revealed leading to a Q-value of action l , $\tilde{Q}^{\pi^*}(b, l) = 99$. Therefore, for this problem, Hindsight is only 50/50 as it takes best expected action in hindsight and has no implicit motivation to gain information. In this problem all the MDP-POMDP solvers discussed in section 5.2, will face the same problem. Making them unfit to solve the famous tiger problem. In the next section we demonstrate that MDP-POMDP solvers are unfit to solve multi-resolution, budgeted information gathering through a scaled down version of the problem.

Multi-resolution, Budgeted Information Gathering

Unmanned Aerial Vehicles (UAVs) have the capability to gain height and gather low resolution information at a large scale and the agility to zoom down on relevant regions of the information to gain high resolution information. This ability makes them suitable for locating objects in the environment and gain information about those objects. The UAV only receives a reward if the high resolution information about the object of interest is collected. Since, the action of gaining height and gaining low resolution information only results in reduction of uncertainty and no actual reward, we show it is an informative action and MDP-POMDP solvers are unable to exploit the ability of UAVs to gather information at multiple resolution.

Let us assume the UAV operates in a grid with cells $C = \{c_1, c_2, c_3, \dots, c_9\}$, with an object of interest, a car in cell c_r , where $r \in [1, 9]$, see figure 5.1. The UAV starts at time $t = 0$ from cell c_5 . At time t , the state of the environment is given by the history of cells visited by the UAV and the location of the car c_r , $s_t = \langle c_r, c_{i_1}, c_{i_2}, c_{i_3}, \dots, c_{i_t} \rangle$, where $c_{i_{1:t}} \in C$. There are five possible actions that the UAV can execute in this grid world, $\mathcal{A} = \{north, east, south, west, up\}$. Action $a = north$ moves the UAV to the north of the cell it currently occupies if possible, otherwise it keeps the UAV in the same cell. Similarly, $east, south, west$ actions move the UAV to the east, south or west of cell it currently occupies respectively. Action up results in the UAV gaining height to inspect the presence of cars in all the

cells and returning to the cell it executed up in. Let, $T : \mathcal{S} \times \mathcal{A} \rightarrow \mathcal{C}$ be the deterministic transition function that takes the state of the UAV and an action as input and returns the the cell in which the UAV will be after the action is executed.

In the MDP definition of the problem, location of the car is known. The UAV gets a high reward for going to the same cell as the car, all actions cost -1 and action up costs relatively higher, -2 . The reward function is given by equation 5.18.

$$R(s, a) = \begin{cases} 100 & \text{if } T(s, a) = c_r \wedge \nexists c_{i_j} \in \mathcal{S} : c_{i_j} = c_r, \\ -2 & \text{if } a = up, \\ -1 & \text{otherwise.} \end{cases} \quad (5.18)$$

Terminal state is reached when either cell c_5 is visited twice or total cost of taking actions (excluding reward of visiting the same cell as the car) exceeds five.

Since the location of the car is known, an optimal MDP solver will directly approach the car taking actions either *east*, *west*, *north* or *south* and return to the terminal cell c_5 , the value of the starting state, $s = \langle c_r, c_5 \rangle$ is given by equation 5.19.

$$V^{\pi_M^*}(s = c_5) = \begin{cases} 99 & \text{if } c_r \in \{c_2, c_4, c_6, c_8\}, \\ 97 & \text{if } c_r \in \{c_1, c_3, c_7, c_9\}. \end{cases} \quad (5.19)$$

Action up does not offer any reward and increases the cost of all paths that leads to any rewarding state. Therefore, an MDP solver will never take the action up . Action up here is an informative action according definition 3. Presence of an informative action cautions us against using MDP based Approximate POMDP solvers for this problem.

In the POMDP version of the problem, the UAV is uncertain about the location of the car. Assuming ties between actions are broken randomly, and the belief about the location of the car is uniformly distributed, the value of initial belief state $b_0^{uniform}$ given hindsight optimization is used at every step can be computed using equation 5.5 and is given by equation 5.20.

$$\tilde{V}^{\pi_h}(b_0^{uniform}) = 34.125 \quad (5.20)$$

where, π_h is the policy attained by iteratively using hindsight optimization.

The value of information about the car is given by the difference between the expected value if the location of the car is known, equation 5.19, the expected value if it is unknown, equation 5.20, 63.875

The optimal POMDP policy will use the action up to reduce the uncertainty and guarantee that it will find the car in the grid, leading to a value of 96. As per theorem 2 the MDP based solvers are sub-optimal by the sum of the expected value of information and the reward achieved by the informative action, 61.875.

As is evident, MDP-POMDP solvers perform sub-optimally at the budgeted, multi-resolution, information gathering task and fail to leverage the agility of UAVs, making them unsuitable for such applications.

5.5 Summary

The main contributions presented in this chapter are -

- *Identification of informative actions as a symptom of unsuitability of MDP based solvers* - We prove that presence of informative actions with a positive expected value of information, section 5.3, is symptomatic of unsuitability of MDP-POMDP solvers for solving a POMDP. Identification of such actions without the need to solve the full POMDP, can help avoid implementation effort to discover the unsuitability of MDP-POMDP solvers and make better design choices.
- *Demonstrating Unsuitability of MDP based solvers for multi-resolution data gathering* - We demonstrated that capability of UAVs to gain height and acquire large-scale, low resolution information is not exploited by MDP based POMDP solvers, hence we need to model the belief space dynamics to enable efficient multi-resolution data gathering, section 5.4.

Contributions towards Enabling Safe Data Gathering in Physical Spaces

In this work we demonstrated that efficient data gathering agents in physical spaces should be able reason about their capabilities to move in the environment and gather data at multiple resolutions. This behaviour can be achieved if solvers for vehicle motion planning for active data gathering reason about motion constraints and belief space dynamics.

Since, decision-theoretic solvers have to operate in a high dimensional belief space and account for all possible future observations, they are not suitable for online implementation. The next chapter presents a method to enable planning for budgeted, multi-resolution data gathering online in diverse environments.

6

Randomized Algorithm for Path Planning for Budgeted Data Gathering

In the last chapter we demonstrated that for solvers to generate near-optimal solution to multi-resolution, budgeted data gathering problems, they need to model high-dimensional belief space dynamics. In this chapter we describe a solver that enables generation of near-optimal solutions to the multi-resolution, budgeted data gathering problem online.

Consider a situation where a region, affected by floods. There is an urgent need to locate survivors, provide supplies and establish communication with them. We have at our disposal an UAV with a camera that can fly up to a limited distance of 1 km. Then the problem becomes that of locating and counting as many survivors as possible while being restricted to fly 1 km. The robot needs to reason about the fact that it can gain a lot of information about the area by gaining height and subsequently flying close to survivors to get a more detailed picture.

Thus the two salient characteristics of the problem are

1. *Constraint on the total travel distance.* Due to limitations of fuel / battery, the length of the route taken by the robot is limited.
2. *Correlated nature of information.* Equipped with cameras, UAVs can view large areas from a distance to gain information. This leads to reward of visiting different locations being correlated.

The problem of planning routes to gain information is an NP-hard problem [Krause, 2008]. Moreover, in the last chapter we proved that in order to find efficient solutions to the problem, planner has to take high-dimensional belief space dynamics into account. The current state of the art systems attempt to solve this problem using two lines of approach. One approach is to apply a myopic information theoretic strategy [Yamauchi, 1997, Charrow et al., 2015] where a set of sensing locations is identified and the system travels to the most promising one. While such strategies are computationally efficient, they fail to effectively account for the constraint on traveling distance. As a result the computed routes can lead to oscillatory behavior.

Another approach is to invoke a long horizon planner [Yu et al., 2014, Singh et al., 2009, CHEN et al., 2014, Zhang and Vorobeychik, 2016].

However, these approaches are far from real-time.

The key contribution of this chapter is an anytime, provably optimal algorithm, Randomized Anytime Orienteering (RAOr), that can efficiently solve for routes that maximize correlated reward functions subject to constraints on route length. The key ideas behind RAOr is to restrict the search space to routes that incur minimum distance to visit selected nodes, and rapidly search this space using random sampling.

We provide empirical and theoretical analysis of RAOr. We prove it to be asymptotically optimal while providing convergence rate analysis. Empirically the algorithm outperforms the state of the art by more than an order of magnitude in terms of run time required to solve benchmark problems near optimally.

In the next Section we formulate the problem of budgeted, multi-resolution information gathering and describe the essential related work. In Section 6.2 we elucidate on transforming Correlated Orienteering to a Constraint Satisfaction Problem and a Traveling Salesman Problem. The resultant algorithm (RAOr, Alg. 8) obtained by leveraging the new formulation is described in Section 6.3 with its properties and drawbacks. These drawbacks are addressed in Section 6.4. The evaluation results and conclusions along with scope of future work is presented in Section 6.8.

6.1 Information Theoretic Formulation for Explicit Data Gathering

Let us formally define the problem of maximizing the reward in a given experiment while respecting a traveling budget B . Let $V = [v_1, v_2, v_3, \dots, v_n]$ be the set of all sensing location in the workspace. Let the robot start from node $v_s \in V$ and end at node $v_e \in V$. Let $r = [v_s, v_1^r, \dots, v_m^r, v_e]$ be an ordered set of the sensing locations where, $r : r \subseteq V$. For each r , let $I(r)$ be the reward gained by visiting each location in P . Let the cost of traversal be given by $C(r) = \sum_{i=1:|r|-1} C(v_i^r, v_{i+1}^r)$, where v_i is the i^{th} element in r , $\forall i \in [1, |r|]$. The problem then is defined by equation 6.1.

$$\arg \max_{r \subseteq V} I(r) \text{ subject to } C(r) \leq B \quad (6.1)$$

The Orienteering Problem (OP) [Golden et al., 1987] work is closely related to the exploration problem. The Orienteering Problem defines the reward at nodes to be independent of each other or in other words the the reward function is modular. Constant approximation ratio of $(2 + \epsilon)$ were given by [Chekuri et al., 2012], but such guarantee is unsatisfactory in practice. On the other hand Mixed Integer Programing(MIP) based solutions exist for OP and related problems [Vansteenwegen et al., 2011]. But these

solutions fail to capture the reward relationship amongst nodes, leading to sub-optimal paths.

[Krause, 2008] established that information gain is sub-modular and monotonic. Nemhauser in 1978, [Nemhauser et al., 1978] provided an efficient method to optimize submodular functions. Unfortunately adding the traveling budget constraint makes the problem non-submodular and non-monotonic. [Chekuri and Pal, 2005, Singh et al., 2009] suggested using a *recursive-greedy* to find approximate solutions for the orienteering problem if the reward function is submodular. Due to large run-times none of these solutions scale well to real world problems. The runtimes of most of the algorithms exceed 3 minutes on a standard desktop PC for a graph of more than 100 nodes.

The MIP based methods solve for linearly relaxed versions of the problem and then impose integer constraints. This results in the method spending most of its time finding partial solutions that do not meet the budget constraints. Also, the state of the art algorithms optimize for the nodes to visit and the sequence in which to travel those nodes together. This leads to the algorithms searching a huge space of solutions for which the sequence of nodes traversed is sub-optimal.

The large run times and the failure to model the reward relationships amongst nodes, necessitates the development of better exploration planning algorithms for efficient autonomous information gathering systems. In the following section we propose reformulating the problem as a combination of Constraint Satisfaction and Traveling Salesman Problem to overcome these limitations.

6.2 From Data Gathering to Orienteering to Set Selection and TSP

The solution to the correlated orienteering problem is a route in a graph, such that the reward attained by the route is maximized while the path cost stays within a specified value. We break the problem of finding the route, into finding the set of locations to visit and then finding the optimal order in which to visit those locations. Since, we assume the reward function is independent of the order in which the set is visited. This allows the set selection and set order optimization to run independently, without affecting the reward attained by a set.

The set selection problem can be formulated as a Constraint Satisfaction Problem (CSP) and finding the optimal order for the selected set is studied as the Traveling Salesman Problem (TSP). In the following we pose the correlated orienteering as a combination of CSP and TSP. The resulting algorithm (RAOr) is presented in Alg. 8 and described in section 6.3

Algorithm 8: Randomized Anytime Orienteering(RAOr)

```

Input:  $G = [V, E], v_s, v_e, B, T_r$ 
Output: The best route found in run-time  $T_r$ 
 $S = \text{SampleSet}(V)$  // Random set is picked such that it contains
 $v_s$  and  $v_e$ 
 $r = \text{TSP}(S, v_s, v_e)$ 
 $r_{best} = \phi$ 
if  $\text{RouteLength}(r) \leq B \wedge \text{Reward}(r) > \text{Reward}(r_b)$  then
  |  $r_{best} = r$ 
end
for  $i = 1 : 3|V|$  do
  |  $v_{new} = \text{Sample}(V)$  // sample a node
  | if  $\text{IsInRoute}(r, v_{new})$  then
  | |  $r = \text{DeleteFromRoute}(r, v_{new})$ 
  | end
  | else
  | |  $r = \text{AddToRoute}(r, v_{new})$ 
  | end
  | if  $\text{RouteLength}(r) \leq B \wedge \text{Reward}(r) > \text{Reward}(r_{best})$  then
  | |  $r_{best} = r$ 
  | end
end
return  $r$ 

```

Constraint Satisfaction Problem

In this section we describe how the Correlated Orienteering problem can be viewed as a Constraint Satisfaction Problem. Let r^* be the solution to equation 6.1. Let, V^* be the set of nodes that are present in r^* . Let, $a_r \in \{0, 1\}^{|V|}$ signify the presence of the nodes in a route r , such that $a_r^i = 1$ if $v_i \in r$ and $a_r^i = 0$, otherwise. In order to solve the correlated orienteering problem we want to find the assignment a_{r^*} and then the optimal order in which nodes belonging to r^* , should be visited.

Randomized technique for efficient search for satisfying assignment of a binary tuple was presented in 1999 in [Schöning, 1999a] as a solution to the CSP problem. RAOr employs the same technique to search for the optimal assignment of $a = a_{r^*}$, i.e. randomly flipping the assignment of one of $|V|$ bits of a .

Traveling Salesman Problem

Once the correct set of nodes are found, then finding the optimal route just requires finding the order in which they need to be visited. Traveling Salesman Problem solvers can provide us with a near-optimal order in polynomial time, [Christofides, 1976b]. Hence combining the TSP and CSP solvers allows us to develop an algorithm to solve the correlated orienteering problem near optimally and efficiently.

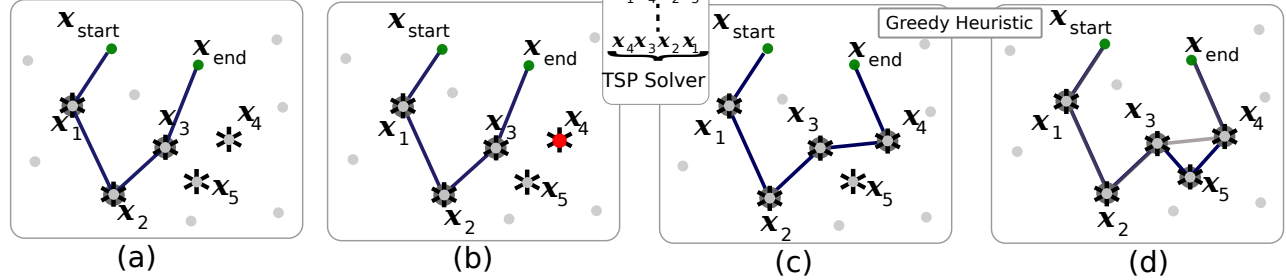
6.3 Randomized Anytime Orienteering

In this section we describe Alg. 8 in detail. We highlight its properties and drawbacks with examples. In the next section we will discuss methods to

overcome these drawbacks.

* High Value Node

• Low Value Node



The algorithm starts with uniformly sampling a set of nodes $S \subset V | v_s, v_e \in S$ (Alg. 8 Line 1). The order in which S should be visited is computed using a TSP solver (Alg. 8 Line 2). Running a TSP solver searches an exponential space of routes in polynomial time, leading to run time reduction. The generated route is then checked for satisfying the budget constraint and saved as the best available route if it is admissible (Alg.8 Line 2-3).

The algorithm then uniformly samples a node $v_{new} \in V$ and changes its status of being in route r (Alg. 8 Line 6-12). i.e if the node v_{new} was in route r , it is removed from route r or if it was not in route r , it is added to it. The new route obtained is checked for satisfying the budget and saved as best available route, if it exceeds the value of r_{best} . This process is repeated $3|V|$ times.

The algorithm described above is the standard probabilistic algorithm for constraint satisfaction problem as suggested in [Schöning, 1999b] changed to return the best route available in the budget. We now list some of the properties and drawbacks of the algorithm.

Theorem 3 (Optimality of Randomized Anytime Orienteering). *Randomized Anytime Orienteering algorithm almost surely finds the optimal route from start to end nodes, within budget B , if there exists one, within a polynomial factor of $2(1 - 1/|V|)^{|V|}$ repetitions.*

The proof of the theorem is provided in section 6.7.

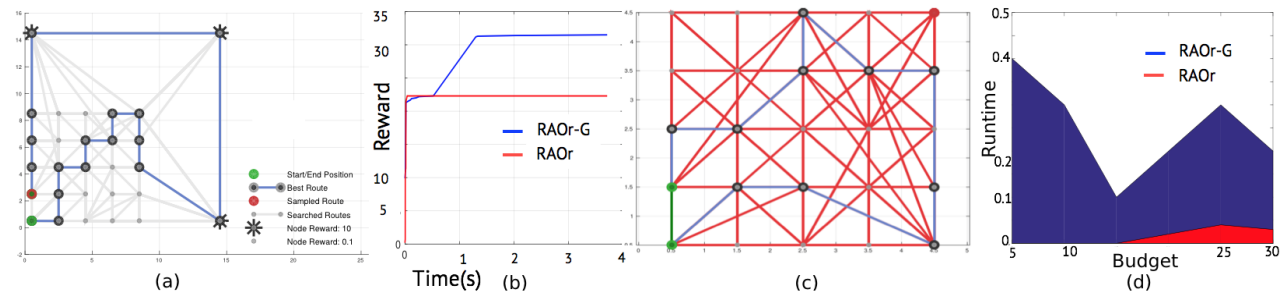
Finding the optimal route for the TSP is an NP-Hard in a space that is exponential in number of nodes. But, polynomial time α approximation algorithms, where $\alpha \geq 1$, for a TSP exist in literature [Christofides, 1976a]. We leverage the polynomial time TSP solver to make Alg. 8 tractable.

Theorem 4 (α -Optimality of Randomized Anytime Orienteering). *Randomized Anytime Orienteering algorithm almost surely finds the optimal route from start to end nodes, within budget B/α , if there exists one, within a polynomial factor of $2(1 - 1/|V|)^{|V|}$ repetitions. Given that the TSP solver in the*

Figure 6.1: An illustration how RAO-G combines CSP and TSP solvers in combination with local greedy heuristics to explore the space of routes rapidly, resulting in improvement of run times for finding near optimal solutions for the correlated orienteering problem. a) Using the CSP algorithm the current admissible route is given by $r = x_{start}, x_1, x_2, x_3, x_{end}$. b) At the next step x_4 is sampled, and is to be added to the route. c) The exponential number of ways in which x_4 can be added to r is reduced to a near optimal order in polynomial time by the TSP solver. This step reduces an exponential search space with in polynomial computation costs. d) The new route obtained is then improved by conducting a local search using greedy heuristics.

inner loop is α -approximate.

Drawbacks



Uniform Sampling of Sets

RAOr has low run-time for the case where a large number of routes lie within the budget and perform near optimally. But the problem scenarios where the probability measure of the optimal path is small the runtime of the algorithm is unacceptable. Fig. 6.2 (a) and (b) presents a problem where the route has to pass through the three high value nodes to achieve close to optimal reward, while the budget is just sufficient to do so. RAOr needs to be able sample from a highly restricted set of feasible near optimal sets. This leads to large run times.

Sampling Inadmissible Sets

RAOr can potentially spend a lot of time considering sets that are out of budget if budget is small as compared to graph size. As is demonstrated in Fig. 6.2 (c) and (d).

To alleviate the drawbacks of the problem we improve the algorithm such that it still keeps its global solution finding properties but improve its convergence properties in these pathological cases. These improvements are defined in the next section and the improved algorithm is described in algorithm 9.

6.4 Randomized Anytime Orienteering - Greedy (RAOr-G)

RAOr (algorithm 8) uniformly samples in the space of sets to find the optimal set of nodes that should be in route. This provides with global solution optimality guarantees while sacrificing on run-times for some pathological cases.

We augment the algorithm to improve its anytime properties by leveraging the problem structure in the following ways -

Figure 6.2: Illustration of drawbacks of RAOr algorithm. a) The problem shown in here consists of three high valued nodes and a budget just sufficient to visit all three. The optimal solution is to visit all three nodes but very few routes exist that have a reward close to optimal. The solution shown is found by running RAOr-G algorithm, Alg. 9. b) Shows the run time vs reward plot for the problem shown in (a) for both RAOr-G and RAOr. RAOr can only attain approximately 66% of the optimal value, as it has a hard time selecting the correct set of nodes. c) Illustrates the problem RAOr faces while running on problems with a small budget. Here a 5X5 grid of nodes, distributed uniformly at a resolution of 1 with a budget of 15 is shown. The routes in red are routes sampled by RAOr that exceed the budget, whereas routes in blue are routes that were sampled that do not exceed the budget. Visibly, red routes outnumber the blue routes. d) The same problem leads to poor run times of RAOr for relatively low budget problems. Here we show the run times of RAOr and RAOr-G on the 5X5 grid shown in (c) while varying the available budget. A runtime of 0 signifies that the algorithm did not converge. Clearly RAOr did not converge for low budget problems in the allotted time. Highlighting the limitation of RAOr in dealing with low budget problems.

Algorithm 9: RAOr-Greedy(RAOr-G)

```

Input:  $G = [V, E], v_s, v_e, B, T_r$ 
Output: The best route found in run-time  $T_r$ 
 $r_g = TSP(v_s, v_e)$  // Seed local search if no global solution is
found
 $R = r_g$ 
if  $RouteLength(r_g) > B$  then
  | return  $\emptyset$ 
end
 $S = SampleSet(V, v_s, v_e)$ 
 $r = TSP(S, v_s, v_e)$ 
if  $RouteLength(r) \leq B$  then
  |  $R = R \cup r$ 
end
for  $i = 1 : \lceil 3|V| \wedge runtime < T_r$  do
  |  $v_{new} = WeightedSample(V)$ 
  | if  $IsInRoute(r, v_{new})$  then
  | |  $r = DeleteFromRoute(r, v_{new})$ 
  | end
  | else
  | |  $r = AddToRoute(r, v_{new})$ 
  | end
  | if  $RouteLength(r) \leq B$  then
  | |  $R = R \cup r$ 
  | end
  |  $R = GreedyLocalSearch(G, v_{new}, R, B)$ 
end
return  $\arg \max_{r \in R} I(R)$ 

```

1. Conducting local searches in the space of routes.
2. Restricting the local search to admissible sets.
3. Informed sampling to improve the likelihood of sampling high value nodes.

The resulting algorithm is called Randomized Anytime Orienteering - Greedy (RAOr-G) and is presented in Alg. 9, Fig. 6.1.

Since, RAOr-G (Alg. 9) is very similar to RAOr (Alg. 8) we highlight the differences here. RAOr-G is seeded with a route which consists of only v_s, v_e , Alg. 9, Line 1-2. The set of nodes is sampled as in RAOr (Alg. 8) but if the resulting route generated by running TSP on selected nodes lies within budget it is added to the set of feasible routes R , Alg. 9, Line 8-10. R is then used by the local search algorithm as the candidate set for running greedy local search, Alg. 9, Line 22. The *GreedyLocalSearch* function is presented in Alg 10 and described in Section 6.4. Another major difference is that instead of uniformly sampling for the node to add or delete from the route, the sampling is weighted in order to sample more valuable nodes often. The intuition behind weighted sampling is described in Section 6.4.

Local Greedy Search Heuristic

Local greedy search heuristic is used to improve the runtime of RAOr algorithm and also to improve the anytime performance of the algorithm. The

Algorithm 10: GreedyLocalSearch

```

Input:  $G = [V, E], v_{new}, R, B$ 
Output: Updated set  $R$  after conducting local greedy search
 $r_c = \text{FindBestRouteInBudget}(v_{new}, R)$ 
if  $r_c == \emptyset$  then
   $r_{cn} = \text{Route}(v_c, v_{new}, v_{new})$  // TSP
end
else
   $r_{cn} = \text{AddToBestRoute}(v_{new}, r_c)$  // Run a TSP with  $v_{new}$  added to  $r_c$ .
end
 $R = R - r_c$ 
 $r_{cn} = \text{AddNearByNodes}(G, r_{cn}, B, d, c)$ 
 $R = R \cup r_{cn}$ 
return  $R$ 

```

greedy search takes as input a set of feasible routes found so far, R and the sampled node v_{new} . It finds the route $r_c \in R$ according to equation 6.2.

$$r_c = \arg \max_{r \in R} \frac{I(r \cup v_{new}) - I(r)}{\text{RouteLength}(r \cup v_{new})} \quad (6.2)$$

$$\text{subject to } \text{RouteLength}(r \cup v_{new}) \leq B$$

The reward of the updated route is further improved by adding nodes that are within distance d and increased the reward gained by the route by atleast c , while keeping it in budget 11.

Algorithm 11: AddNearByNodes

```

Input:  $G = [V, E], r_{cn}, B, d, c$ 
Output: Updated set  $R$  after conducting local greedy search
forall the  $v \in V | \text{DistanceFromRoute}(v, r_{cn}) \leq d$  do
  if  $(I(r_{cn} \cup v) - I(r_{cn})) \geq c$  then
    if  $\text{DistanceFromRoute}(v, r_{cn}) * 2 \leq B - \text{RouteLength}(r_{cn})$  then
       $r_{cn} = \text{AddToRoute}(v, r_{cn})$  // Run a TSP with  $v$  added to  $r_{cn}$ .
    end
  end
end
return  $r_{cn}$ 

```

The total computation cost of a single iteration of local greedy search heuristic is $O(|R||V|)$. In practice we have found that the speedup achieved far outweighs the cost, Fig. 6.2.

In the next Section we describe the intuition behind using weighted sampling instead of uniform sampling and how it affects the theoretical guarantees provided by the RAO-G algorithm (Alg 9).

Weighted Sampling

Weighted sampling, samples nodes with the probability directly proportional to the reward they offer independently. The intuition behind weighted sampling is that nodes with high values tend to be the part of optimal routes. This

intuition is very similar to that of constructing macro-actions described in [He et al., 2010]. Unfortunately weighted sampling adversely affects the theoretical guarantees of the algorithm, but empirically it leads to alleviating the problems caused by uniform sampling 6.3.

Theorem 5 (Optimality of Randomized Anytime Orienteering - Greedy).

Randomized Anytime Orienteering - Greedy algorithm almost surely finds the optimal route from start to end nodes, within budget B/α , if there exists one in polynomial factor of $\left(\frac{2}{1+\zeta}\right)^{|V|}$ repetitions, where $\zeta = \left(\frac{((|V|-I_{min})I_{min})^\beta}{(|V|-1)^{(\beta+1)}}\right)$,

$$I_{min} = \min_{v \in V \text{ and } I(v) \neq 0} I(v), \beta = \frac{1}{|V|-2}.$$

6.5 Results

We evaluate the computational performance of RAO_r on various benchmark examples against the state of the art methods. Then, we apply the algorithm on a realistic coverage scenario in simulation. All simulation computations are performed on a computer equipped with Intel Core i7-4870HQ using Matlab. The algorithm was also deployed on an autonomous exploration UAV system, we present and analyze multiple exploration flights.

Computational Performance

The benchmark problem, Fig 6.3 consists of a graph with nodes located in a grid at uniform resolution. The reward for visiting each cell in the grid is 1. There is no reward for visiting a cell twice. Table 6.1 shows the comparative run times of state of the art, near-optimal algorithms vs RAO_r-G for different problem sizes and varying budgets. MIQP [Vansteenwegen et al., 2011] is a fixed time algorithm, which return after finding a near-optimal solution. Both RAO_r-G and eSIP [Singh et al., 2009] are anytime in nature. Each algorithm was stopped when it reached 95% of the optimal value. RAO_r-G outperforms the state of the art in all the benchmark problems. RAO_r-G solves the problem with 400 nodes approximately 100 times faster than the eSIP.

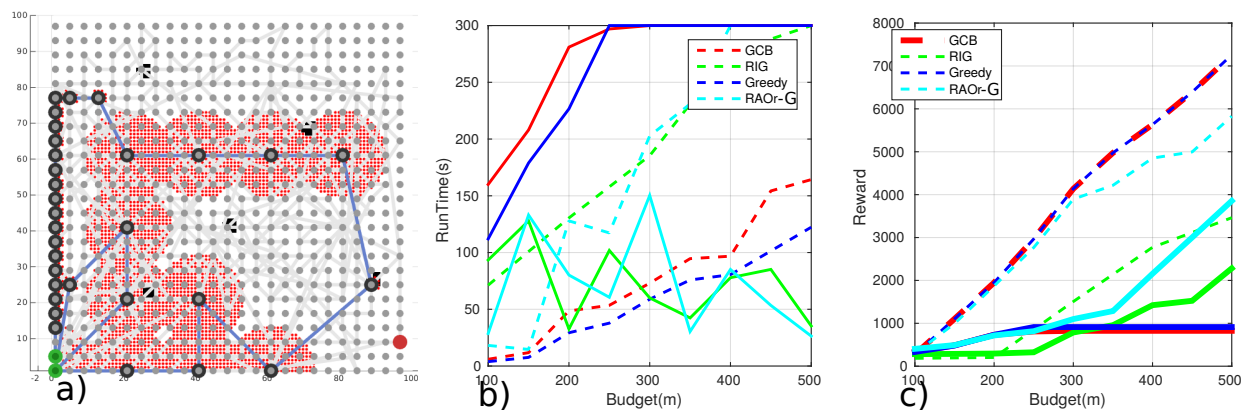


Figure 6.3: Illustrated here is 10X10 grid size benchmark problem. For all the benchmark problems the nodes are situated in a uniform grid with 1 resolution. For this particular problem RAO_r-G is able to find a near-optimal solution in 6.9 seconds while the state of the art takes 143.6 seconds for finding same quality of solution.

Grid Size	Budget		eSIP	MIQP	RAOr-G
5X5	30	cost	29.8	29.6	29.4
		utility	24	25	24
		time(s)	6.8	15.3	0.5
7X7	60	cost	56.6	57.3	56.1
		utility	48	49	48
		time(s)	26.7	1358.5	2.4
10X10	100	cost	99.0	99.8	98.9
		utility	87	92	87
		time(s)	143.6	15330	6.9
20X20	100	cost	99.7	-	99.2
		utility	87	-	87
		time(s)	763.7	-	7.2

Correlated Rewards

In the first scenario we compared the computational performance against algorithms that promise optimality but on a simple case, where rewards of the nodes were independent. Here we evaluate RAO-G's performance against algorithms that can potentially work on-board robots given their relatively low runtimes for about over 100 randomly generated cases.



The algorithm has to find a route given start and end locations such that the reward collected is maximized. Reward at a node is defined by the region visible from a 25° field of view, downward facing camera. Viewpoints exist on grid of $4m$, at a height of $10m$ and at a uniform distance of $20m$ at a height of $55m$. Resulting a total of 650 nodes/viewpoints. The area is randomly strewn with 5 high value objects, covering those from lower viewpoints results in a 10 times higher reward than other locations. Two distance metrics are implemented, Case1 Euclidean distance and Case2

Table 6.1: Comparative run time analysis of state of the art v/s RAO-G. RAO-G consistently outperforms the state of the art in terms of run times for achieving near-optimal solutions. All the solutions obtained are atleast 95% of the optimal.

Figure 6.4: a) RAO-G planned path with a $500m$ budget on the 100×100 area for Case2. Red marks the sensor footprint. Notice how the path visits high value regions, displayed in black. Grey paths show all the paths searched by RAO-G. b) and c) Case 1 is displayed in dashed line, Case 2 in solid lines. RAO-G is competitive with greedy algorithms in Case 1 and dramatically outperforms greedy and RIG for Case 2, where greedy algorithms are stuck in local maxima.

Euclidean distance with travel along z-direction costing 3 times more vs. traveling in x-y plane, see figure 6.4. Since nodes/viewpoints are distributed in a uniform fashion, greedy solutions are able to perform near optimally in Case 1. RAO-G performs comparably to the greedy solutions, while incurring a bit more runtime. In Case 2, greedy solutions are clearly stuck in a local maxima, leading to a much worst performance than RAO-G. In both cases RAO-G outperforms RIG algorithm, [Hollinger], both in runtimes and solution quality, see figure 6.4.

GCB vs RAO-G

Fig. 6.5 illustrates one the runs from *Case2* in the above section. GCB is a non-myopic greedy algorithm for IPP. Since, gaining height is costly, the greedy algorithm GCB is unable gain height leading to poorer performance than RAO-G which non-myopically reasons about the reward available. For the given budget of 500, GCB manages to gain a reward of 1000, whereas RAO-G gains a reward of 4000.

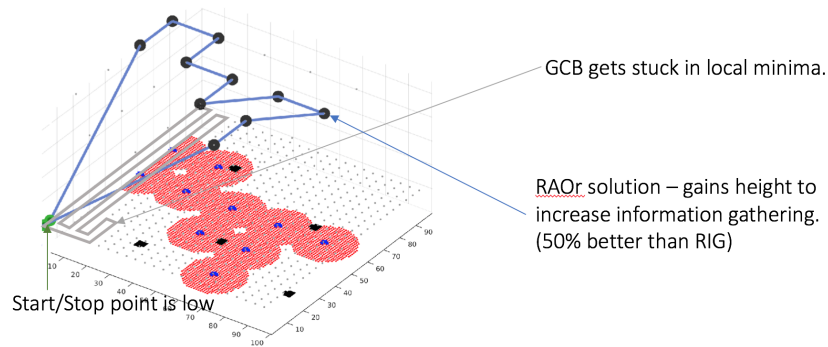


Figure 6.5: Performance comparison of GCB and RAO-G for a scenario where gaining height is expensive and hence greedy algorithms tend to get stuck in local minima.

Case Study 6.1: Exploration mission with low budget

The algorithm was also deployed on the autonomous UAV system described in section 6.8. It ran at 3Hz on an Odroid-C2. Figure 6.6 describes the vehicle mission. The vehicle is deployed to scout for cars and collect high resolution data if a car is found. The exploration reward function is weighted probability distance between current robot's belief and expected updated belief and a fixed reward of 20000 is allocated to scanning a car, for more details about the representation see section 6.8. The algorithm run's adaptively as the vehicle's representation is updated and is able to guide the vehicle to explore the environment and locate and map both the cars, see figure 6.6. Since the vehicle uses RAO-G, a non-myopic planner, it does not rush to the first car it maps in the environment, rather realizes that it will be cheaper to gather the high resolution imagery of the car at the return leg. Saving 32m of cost in he process.

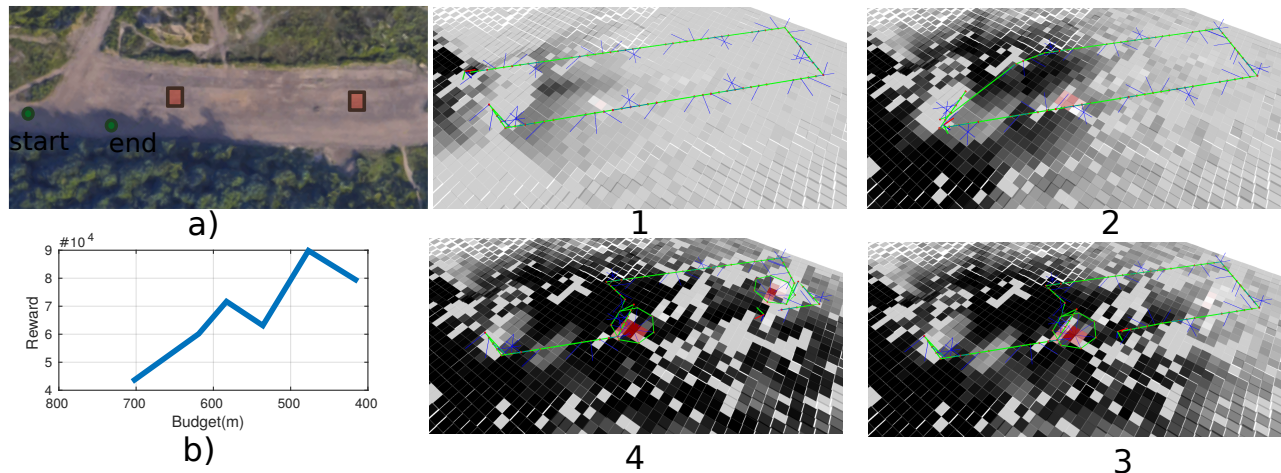


Figure 6.6: a) Testing site, start and end are marked by green nodes and car locations are shown in orange. b) Vehicle starts with a budget of $700m$, the reward increases as likelihood of finding the car increases, the crest in reward marks the time at which the global planner found and decided to map the car. Figures 1, 2, 3 and, 4 show the series of plans at various stages of the exploration mission, Dark squares indicate absence of cars and red squares presence of cars. Shades of grey and red signify certainty. Once the car is recognized, a 360 view of the car is associated with a high reward.

Case Study 6.2: Flight tests at Gascola

Fig. 6.7, shows the largest data gathering mission conducted during field trials. Simulated semantic classification images were used for this trial. The travelling budget of the system was $1.9km$ with approximately $150,000m^2$ area to be covered, the vehicle was able to cover the region in a single battery pack while finding and mapping both the cars hidden in the environment. For a greedy approach to collect the same resolution imagery of the car would have taken over 35 minutes and 7 battery changes.

For repeatability 15 successful missions were conducted, where the robot was tasked to find and map cars at high resolution in Gascola. Fig 6.8 shows the distribution location of cars that the UAV found, as well as the start and end points.

6.6 Summary

In this chapter, we presented the development and evaluation of an informative path planning algorithm that enables the online solution of budgeted, multi-resolution, data gathering problems.

Main contributions of the work are as follows -

- *Asymptotically optimal IPP algorithm*- We introduced RAO an asymptotically optimal information path planning algorithm. That can efficiently solve for routes that maximize correlated reward functions subject to constraints on route length. The key insight in the problem structure is that the order in which viewpoints are visited in the world is independent of the reward gained by visiting them. This property enables the problem to be broken into the selection of nodes to visit and the order in which to visit them, leading to an exponential reduction in effective search space.

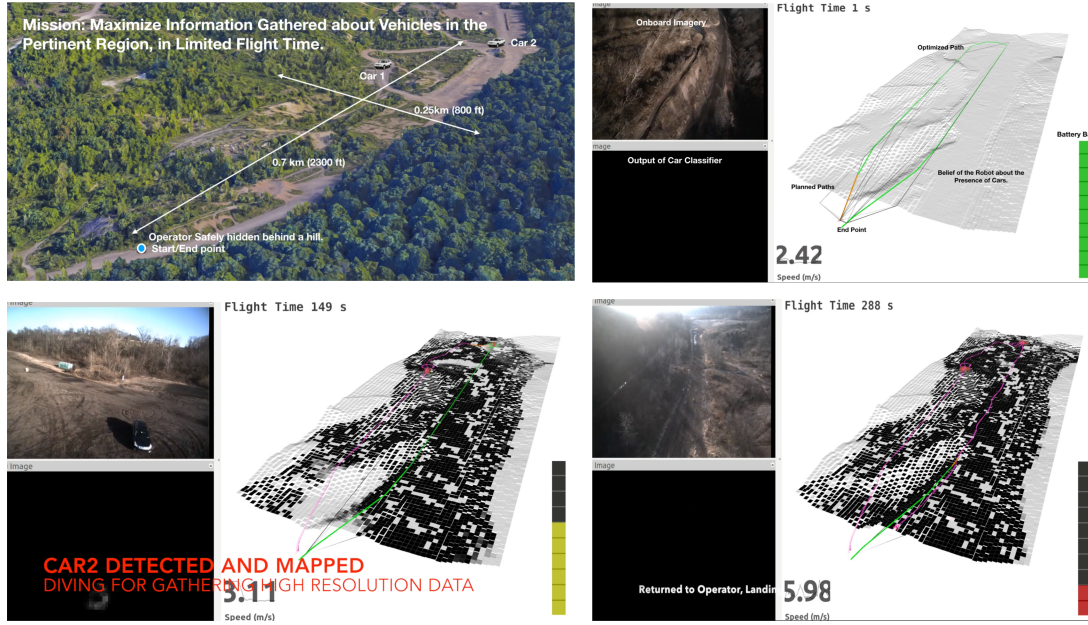


Figure 6.7: Long-range exploration mission, where the robot was tasked to explore $150,000m^2$ of area, and it completed the mission in 1 battery pack, just under 5 minutes. The top left image shows the scenario, and other 3 images show mission progress in chronological order as denoted by the flight time displayed in them.



Figure 6.8: All the locations where the robot has been successfully able to find and map cars in high resolution (Red dots). Start and end points of missions from which the robot began and ended scouting. (White Dots)

- *Experimental validation of RAO_r and Multi-resolution information gathering* - We evaluate RAO_r on a variety of toy, simulation, and real-world scenarios. We show that it performs as well as greedy solvers, where greedy methods are near-optimal while avoiding to get stuck in local minima for problems where greedy approaches are ineffective. We also show that multi-resolution data gathering can significantly increase the effectiveness of UAVs as data gathering tools.

Contributions towards Enabling Safe Data Gathering in Physical Spaces

This work formally defined the budgeted data gathering problem (equation 6.1) with a belief dependent reward function. We then presented an online algorithm that can address budgeted, multi-resolution data gathering problem online. *RAO_r-G* combined with safety algorithms described in chapter 3 and the sensor motion planning algorithm described in chapter 4 form a framework to enable safe, efficient data gatherers for mobile agents whose pose uncertainty can be ignored. The algorithms have focussed on producing online solvers to the data gathering problem that can reason about motion constraints. As a result of modeling motion constraints, we have been able to provide performance guarantees for the techniques described.

Before moving on to the conclusion, we present the proofs, and details of the representations that enabled experiments presented in section 6.5.

6.7 Proofs

Proposition 8 (Returning optimal route on optimal set selection). *If RAO_r finds the optimal set of nodes to visit during its run-time, it is guaranteed to return the corresponding optimal route, if an optimal TSP solver is used.*

Proof. In every iteration, RAO_r (Alg. 8) checks if the set of nodes selected in that iteration can be visited within the budget constraint. The optimal order to visit the selected set is given by a TSP solver. If the selected set can be visited within the budget, it is stored as the best set encountered if it is more rewarding than the previous best set encountered. The most rewarding set encountered in the budget is returned. Therefore, if set V^* is encountered it is guaranteed that r^* will be found and returned, given we have an optimal TSP solver. Hence, if RAO_r finds the optimal set of nodes to visit during its run-time, it is guaranteed to return the corresponding optimal route, if an optimal TSP solver is used. ■

We next compute the probability of finding the optimal assignment or equivalently the optimal path using RAO_r.

Proposition 9 (Probability of finding the correct path). *Randomized Any-time Orienteering algorithm finds the optimal route from start to end nodes, within budget B , if there exists one, with a probability of at least $\left(\frac{1}{2} \left(1 + \frac{1}{|V|-1}\right)\right)^{|V|}$ in one run.*

Proof. The proof of this theorem is taken from the work on probabilistic algorithm for constraint satisfaction problem [Schöning, 1999a].

Given a_{r^*} is the optimal solution, we want estimate the lower bound on the probability that Alg. 8 finds a_{r^*} . Once we have found this success probability p , the expected number of independent repetitions of the procedure until we find the optimal solution is $1/p$.

Now, we calculate p . It is clear that the random variable X that counts the number of bits in which the random assignment a and the fixed assignment a_{r^*} disagree (i. e. the Hamming distance between a and a_{r^*}) is binomially distributed. That is, $Pr(X = j) = \binom{|V|}{j} 2^{-|V|}$. If the system is in state 0, this means, an optimal assignment has been found.

At any given point in the algorithm, if a is not the optimal assignment then there must be atleast one vertex out of $|V|$ that needs to be flipped (included or excluded from the set selection) to reduce the hamming distance of a to a_{r^*} reduces by 1. Selecting the correct vertex would mean that the current state transfers j to transfers to state $j - 1$ with probability at least $1/|V|$, and transfers to state $j + 1$ with probability at most $\frac{|V|-1}{|V|}$. This markov chain is the same as described by [Schöning, 1999a]. We present the proof of value of p here for completeness.

Given that the process has initially transferred into state j , we calculate the probability q_j that the process reaches the absorbing state 0. For this

to happen the process needs at least j steps. We consider the case that the random walk takes $i \leq j$ steps in the "wrong" direction and $i + j$ steps are required toward the "right" direction so that the process stops in state 0 after $j + 2i$ steps. To calculate this probability requires us to calculate the number of paths on a rectangular grid (which represents the possible movements over the Markov chain over the time scale) which transfers the process from state j to state 0 while exactly i steps in the "wrong" direction. Using the ballot theorem from [Feller, 1950], page 73, it can be seen that this number is $\binom{j+2i}{i} \cdot \frac{j}{j+2i}$. Therefore, the probability can be estimated as follows.

$$\begin{aligned} q_j &\geq \sum_{i=0}^j \binom{j}{j+2i} \cdot \frac{j}{j+2i} \cdot \left(\frac{|V|-1}{|V|}\right)^i \cdot \left(\frac{1}{|V|}\right)^{i+j} \\ &\geq \frac{1}{3} \sum_{i=0}^j \binom{j}{j+2i} \cdot \left(\frac{|V|-1}{|V|}\right)^i \cdot \left(\frac{1}{|V|}\right)^{i+j} \end{aligned} \quad (6.3)$$

Further we can lower bound the above sum by its largest term as follows. We use the following fact [Motwani and Raghavan, 2010]. $\binom{n}{\beta n} \sim 2^{h(\beta)n} = \left(\frac{1}{\beta}\right)^{\beta n} \left(\frac{1}{1-\beta}\right)^{(1-\beta)n}$ where $h(\beta) = -\beta \log_2 \beta - (1-\beta) \log_2 (1-\beta)$ is the binary entropy function. In particular, the two functions

$$\binom{(1+2\beta)j}{\beta j} \text{ and } \left[\left(\frac{1+2\beta}{\beta}\right)^\beta \cdot \left(\frac{1+2\beta}{1+\beta}\right)^{1+\beta} \right]^j \quad (6.4)$$

are within polynomial factors of each other. We lower bound the above estimation for q_j by setting $\beta = \frac{1}{|V|-2}$.

$$\begin{aligned} q_j &\geq \frac{1}{3} \sum_{i=0}^j \binom{j}{j+2i} \cdot \left(\frac{|V|-1}{|V|}\right)^i \cdot \left(\frac{1}{|V|}\right)^{i+j} \\ &\geq \left[\left(\frac{1+2\beta}{\beta}\right)^\beta \cdot \left(\frac{1+2\beta}{1+\beta}\right)^{1+\beta} \cdot \left(\frac{|V|-1}{|V|}\right)^\beta \cdot \left(\frac{1}{|V|}\right)^{1+\beta} \right]^j \\ &\quad \text{where } \beta = \frac{1}{|V|-2} \\ &\quad = \left(\frac{1}{|V|-1}\right)^j \end{aligned} \quad (6.5)$$

where the last inequality holds up to some polynomial factor. Therefore, up to some polynomial factor, using the binomial theorem, we obtain the following estimate for success probability p

$$\begin{aligned} p &\geq \left(\frac{1}{2}\right)^{|V|} \sum_{j=0}^{|V|} \binom{|V|}{j} \left(\frac{1}{|V|-1}\right)^j \\ &= \left(\frac{1}{2} \left(1 + \frac{1}{|V|-1}\right)\right)^{|V|} \end{aligned} \quad (6.6)$$

■

Theorem 1 (Optimality of Randomized Anytime Orienteering). *Randomized Anytime Orienteering algorithm almost surely finds the optimal route from start to end nodes, within budget B , if there exists one, within a polynomial factor of $2(1 - 1/|V|)^{|V|}$ repetitions.*

Proof. According to Proposition 9 the RAO algorithm finds the optimal set with probability atleast $p \geq \left(\frac{1}{2}\right)^{|V|} \sum_{j=0}^{|V|} \binom{|V|}{j} \left(\frac{1}{|V|-1}\right)^j = \left(\frac{1}{2} \left(1 + \frac{1}{|V|-1}\right)\right)^{|V|}$. The expected number of independent repetitions of the RAO algorithm are $1/p$. The probability that RAO will not find a satisfying assignment after h repetitions is at most $(1 - p)^h \leq e^{-ph}$. Therefore, to achieve acceptable chance of missing the optimal path, say e^{-50} , we can choose $h = 50/p$. So the complexity is within a polynomial factor of $1/p$. Therefore, Randomized Anytime Orienteering algorithm almost surely finds the optimal route from start to end nodes, within budget B , if there exists one, within a polynomial factor of $(2(1 - 1/|V|))^{|V|}$ repetitions. ■

Theorem 2 (α -Optimality of Randomized Anytime Orienteering Algorithm). *Randomized Anytime Orienteering algorithm almost surely finds the optimal route from start to end nodes, within budget B/α , if there exists one, within a polynomial factor of $(2(1 - 1/|V|))^{|V|}$ repetitions. If the TSP solver in the inner loop is α -approximate.*

Proof. If the TSP solver is α optimal, if V^* is encountered during set selection, the length of the route covering the set may not fit inside the budget constraint.

Although, if V_b^* is the most rewarding set that can be visited in $\frac{B}{\alpha}$, then it can be guaranteed that if V_b^* set is selected, it will be returned as the most rewarding set by RAO. The probability of finding V_b^* is presented in equation 6.6 and hence the RAO algorithm almost surely finds the optimal route from start to end nodes, within budget B/α , if there exists one, within a polynomial factor of $(2(1 - 1/|V|))^{|V|}$ repetitions. If the TSP solver in the inner loop is α -approximate. ■

Theorem 3 (Optimality of Randomized Anytime Orienteering - Greedy). *Randomized Anytime Orienteering - Greedy algorithm almost surely finds the optimal route from start to end nodes, within budget B/α , if there exists one in polynomial factor of $\left(\frac{2}{1+\zeta}\right)^{|V|}$ repetitions, where $\zeta = \left(\frac{((|V|-I_{min})I_{min})^\beta}{(|V|-1)^{(\beta+1)}}\right)$,*

$$I_{min} = \min_{v \in V \text{ and } I(v) \neq 0} I(v), \beta = \frac{1}{|V|-2}.$$

Proof. RAO - G relies on the same process as RAO for its global convergence characteristics. The random variable X that counts the number of bits in which the random assignment a_r and the fixed assignment a_{r^*} disagree (i. e. the Hamming distance between a and a_{r^*}) is binomially distributed. That is, $Pr(X = j) = \binom{|V|}{j} 2^{-|V|}$. If the system is in state 0, this means, an optimal assignment has been found.

At any given point in the algorithm, if a is not the optimal assignment then there must be atleast one vertex out of $|V|$ that needs to be flipped to reduce the hamming distance of a to a_{r^*} reduces by 1. Selecting the correct vertex would mean that the current state transfers j to transfers to state $j - 1$ with probability at least $\frac{I_{min}}{|V|}$ and transfers to state $j + 1$ with probability at most $1 - \frac{I_{min}}{|V|}$. The change in probabilities is due to the weighted sampling of nodes instead of uniform sampling.

Substituting these values in equation 6.5 and following the algebra we get the following.

$$q_j \geq \left(\frac{((|V| - I_{min})I_{min})^\beta}{(|V| - 1)^{(\beta+1)}} \right)^j = \zeta^j \quad (6.7)$$

where the inequality holds up to some polynomial factor. Therefore, up to some polynomial factor, using the binomial theorem, we obtain the following estimate for success probability p

$$p \geq \left(\frac{1}{2}\right)^{|V|} \sum_{j=0}^{|V|} \binom{|V|}{j} \zeta^j = \left(\frac{1}{2}(1 + \zeta)\right)^{|V|} \quad (6.8)$$

Hence the number of repetitions required for $RAOr - G$ to find the optimal route is given by a polynomial factor of $\left(\frac{2}{1+\zeta}\right)^{|V|}$. Where ζ is given by $\left(\frac{((|V| - I_{min})I_{min})^\beta}{(|V| - 1)^{(\beta+1)}}\right)$. ■

6.8 Representations & Systems

Micro-Aerial Vehicle Platforms

RAOr-G was tested on multiple Micro-Aerial Vehicles (MAVs) over the course of last 3 years of the project, table 6.2. Each multi-rotor is equipped with a stereo pair, GPS/INS, and a high resolution camera to support semantic segmentation. The base platform is an off-the-shelf quadrotor vehicle retrofitted with in-house developed sensing and computing suite designed for semantic exploration. The sensor suite consists of a monochrome stereo camera pair, a monocular color camera, an integrated GPS/INS unit and a barometer. The stereo camera pair provides 640×480 resolution disparity image at 10 fps for the obstacle avoidance and 3D mapping systems. The central camera is operated at 3 fps, to provide high resolution color imagery for the semantic perception system. All cameras are forward-facing, tilted downwards at 15° , an orientation well suited for low-altitude ($< 40m$) operation. The GPS/INS system and the barometer are used for state estimation, figure 6.9.

All computation for autonomous operation is performed on-board. To this end we equip the MAV with two embedded ARM computers; one of them is

devoted primarily to planning tasks, while the other is devoted to perceptual tasks. In addition, we use a specialized FPGA processor [ner] for stereo depth computation. The computers are networked through high-speed ethernet. We present the representations developed to enable autonomous data gathering using these multi-rotors.


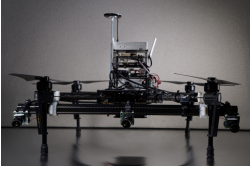

Characteristics	Mikrokopter Octocopter	DJI M100	DJI M100
			
Project year	Year 1	Year 2 and 3	Year 4
Number of propellers	8	4	4
Sensing Payload	2.5 kg	1.2 kg	0.5 kg
Flight time	300 s	300s	480s

Table 6.2: Quadrotor Platforms Comparison

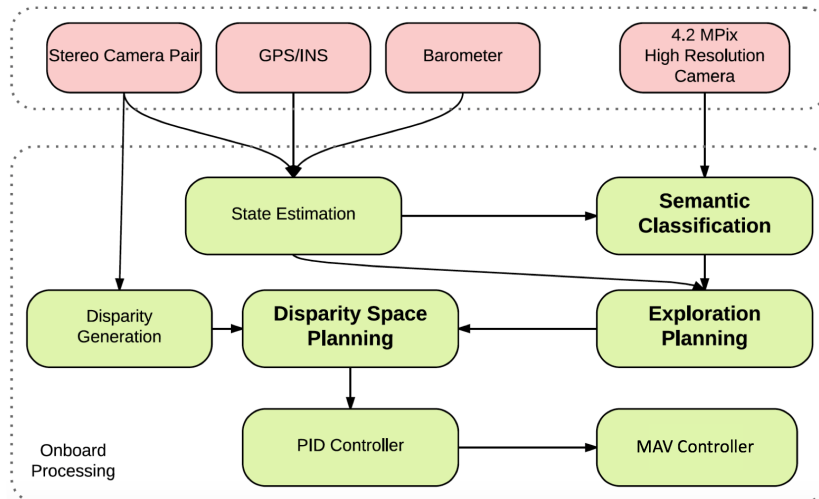


Figure 6.9: MAV System Block Diagram: Exploration Planner *RAOr* provides global trajectories to the disparity planner described in section 6.8. Semantic classification outputs classified images to the exploration planning node, that processes these images to construct a semantic grid map also described in section 6.8

Semantic Grid Representation

The goal of the Semantic Mapping system is to inform the planning system about the presence and approximate location of the classes of interest in its surroundings, so it can create information-gathering plans. It does so by means of a *semantic map*, a metric map that is annotated with localized predictions regarding semantic classes.

Thus, in order to be useful, the system must operate online and in real time, in order to keep the map updated as new sensor data is acquired. Addi-

tionally, it must also be capable of recognizing and localizing distant (up to $200m$) objects, as its function is primarily to help the vehicle decide where to go, and secondarily to describe where it has been.

To this end, the semantic mapping system must answer two questions about the scene: *what* objects of interest are in it, if any, and *where* are they, in physical space. To answer these questions, our semantic mapping module has two main stages. In the first stage, *semantic segmentation*, we use a deep learning system to label monocular camera imagery. In the second stage, *mapping*, we project the segmentation into a 2.5D grid map which maintains the robot’s belief about the semantic class of each grid cell. We describe mapping part of the pipeline in the following section.

Mapping

Given a semantically classified image, we want to find the position of objects detected in the image, as well as model regions for which the information in measurements is uncertain. Since, this mapping has to be performed on board the vehicle, the driving requirement of the application is computation time. Given a global pose by state estimation filter, each pixel in the labeled image defines a ray originating at the camera center and passing through the pixel center. To perform the mapping operation we use the images with soft pixel-wise predictions, together with the robot’s global pose estimate and a pre-existing digital elevation map (DEM). We exploit the semantic knowledge of the world (every object rests on the ground) and use the digital elevation map to infer the 3D structure of the environment.

Minimal computation cost has allowed occupancy grid based mapping algorithms to be successfully deployed on-board robots. The reduced the computational complexity is achieved by reducing the dimensionality of the mapping problem by assuming cells in a grid are independent binary random variables and measurements are independent, given a cell’s true occupancy value. These assumptions have been shown to work effectively with sensors that provide both range and bearing.

But a semantically classified image provides bearing only measurements through rays originating from camera pose, making the ray independence assumption limiting. To fully exploit the bearing only measurement and the semantic structure knowledge of the world, we need to model ray dependence. Section 6.8 and section 6.8 describe how we model dependence amongst observations while still allowing for an on-line mapping algorithm.

Exploiting Semantic Knowledge

We assume that objects of our interest, represented by $\mathcal{L}_{\mathcal{M}} = \{c_1, c_2, \dots, c_n\}$, rest on the ground and we know the likely height $h_{c_i} \forall c_i \in \mathcal{L}_{\mathcal{M}}$. We model the world as a 2.5D grid. In every cell, C_{ij} of the grid at location i, j , we store the heights at which rays pass over the cell for all classes by casting rays

originating from the classified image, table 6.3. We are interested in finding the cells where the height of rays passing over the cell match the height of object we are looking for, while accounting for occlusions and limited field of view. This leads to following cases for a given class in a cell C_{ij} –

Symbol	Description
$C_{ij.c.h_u}$	The highest height at which a ray with label c passes over or intersects the cell C_{ij} .
$C_{ij.c.h_l}$	The lowest height at which a ray with label c passes over or intersects the cell C_{ij} .
$C_{ij.c.n_f}$	The number of rays with label c that pass over or intersect the cell C_{ij} at a height less than h_c .
$C_{ij.c.n_a}$	The number of rays with label other than c pass over or intersect the cell C_{ij} at a height less than h_c .
$C_{ij.c.p_f}$	The cumulative probability of rays with label c that pass over or intersect the cell C_{ij} at a height less than h_c .
$C_{ij.c.p_a}$	The cumulative probability of rays with label other than c that pass over or intersect the cell C_{ij} at a height less than h_c .
$C_{ij.c.l_o}$	Integrated log-odds of an object of class c being present in the cell C_{ij} .

Table 6.3: Data members of grid cell C_{ij} for class c .

- *Case 0*- Average probability of rays that pass over cell C_{ij} with a label other than class c is greater than average probability of rays with class c .
- *Case 1*- Rays of some other class pass from below and above the class of concern over the cell C_{ij} .
- *Case 2*- Rays of some other class pass from below and nothing is observed above the class of concern over the cell C_{ij} .
- *Case 3*- Nothing is observed above or below the class of concern over the cell C_{ij} .
- *Case 4*- Nothing is observed below and some other class is observed above the class of concern over the cell C_{ij} .

Case 1 implies that the cell is well-observed. Therefore, $C_{ij.c.h_u}$ and $C_{ij.c.h_l}$ should be close to h_c and ground height respectively. *Case 2* implies that the upper part of the object could not be sensed due sensing geometry or occlusions. Hence, $C_{ij.c.h_u}$ should be less than h_c and $C_{ij.c.h_l}$ should be close to ground. Similarly, *Case 3* implies that $C_{ij.c.h_u}$ and $C_{ij.c.h_l}$ should be less than h_c and *Case 4* implies that $C_{ij.c.h_u}$ should be close to h_c and $C_{ij.c.h_l}$ should be less than h_c . These cases lead to equation 6.9, that is used to determine whether there is positive, negative or lack of evidence in the

current classified frame regarding the presence of object of class c over the cell C_{ij}

$$\phi_{ij}(c) = \begin{cases} 0, & \text{Case 0} \\ e^{\alpha_k C_{ij}.c.l_1} e^{\beta_k (C_{ij}.c.l_u - h_c) / h_c} \frac{C_{ij}.c.p_f}{C_{ij}.c.n_f}, & \text{Case k} \end{cases} \quad (6.9)$$

Where, $k \in [1, 4]$ and α_k, β_k are negative constants that allow us to change the weights of the measurements according to the cases encountered. We use the following values for these constants, $\alpha_1 = \beta_1 = -10$, $\alpha_2 = -10$, $\beta_2 = -1$, $\alpha_3 = -1$, $\beta_3 = -1$, $\alpha_4 = -1$, $\beta_4 = -10$. Obviously a value of $\phi_{ij}(c)$ close to 0.5 indicates lack of evidence, and $\phi_{ij}(c) < 0.5$ indicates negative evidence and $\phi_{ij}(c) > 0.5$ positive evidence for the presence of class c in cell C_{ij} .

Temporal Evidence Integration

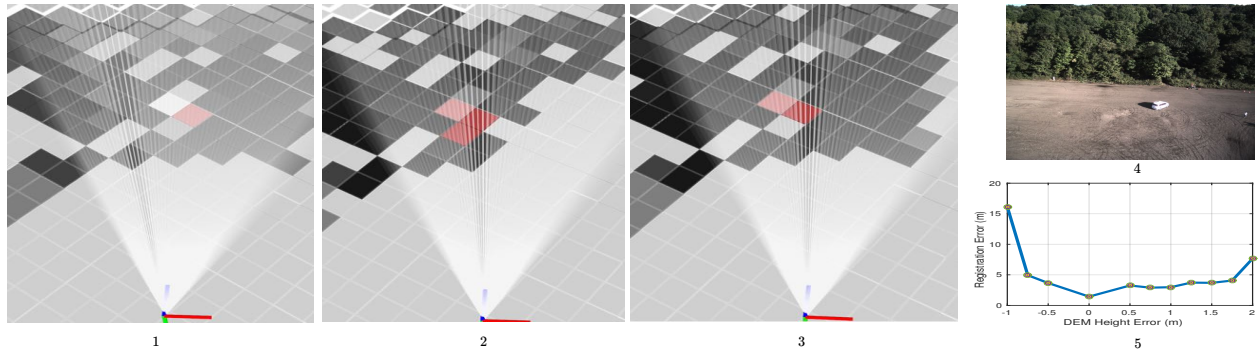
$\phi_{ij}(c)$, allows the algorithm to model the dependence amongst rays, while allowing us to treat the cells independently. We assume at any given cell, the log odds of probability of observing a class c is given by a constant γ . Each class in a cell is represented as an independent binary random variable, as a cell can have objects of multiple classes. Once the nature of evidence ($\phi_{ij}(c)$) is identified, logodds for each class in each cell is updated with equation 6.10.

$$C_{ij}.c.l_o = \begin{cases} C_{ij}.c.l_o, & |\phi_{ij}(c) - 0.5| \leq \zeta \\ C_{ij}.c.l_o + \gamma C_{ij}.c.(n_f - n_a), & \text{otherwise} \end{cases} \quad (6.10)$$

Where, ζ is a small positive number less than 0.5. We use $\zeta = 0.2$ and $\gamma = 1$. Each semantically classified image is integrated with the grid and $C_{ij}.c.l_o$ is updated for every cell that needs updating, this process is repeated for every input semantically classified image. Next section presents the hardware system on which we run the semantic mapping system to enable autonomous scouting. Preliminary results for the mapping algorithm are presented in section 6.8.

Mapping Example

In this section we demonstrate the effects of exploiting semantic knowledge and modeling ray dependence qualitatively, while measuring the sensitivity of the mapping algorithm to height inaccuracies in the DEM. Figure 6.10-4 shows a canonical scenario where a car, more than 50m away, is detected by the semantic classification algorithm. Exploiting semantic knowledge and modelling dependence allows the mapping algorithm to capture the uncertainty about the presence of a car in the cell occluded by the car, figure 6.10-1, whereas if we do not reason about ray interdependence, the occluded



cell is also inferred to contain cars figure 6.10-3. If both the semantic knowledge and ray interdependence are not exploited, then a simple projection of classified image to the DEM leads to an inference that multiple cells are occupied by a car figure 6.10-2. Demonstrating that modelling the ray interdependence and exploiting semantic knowledge leads to better mapping of objects and uncertainties. Figure 6.10 shows that the algorithm's performance deteriorates in presence of height errors in the DEM. Unsurprisingly the degradation is faster if DEM underestimates the height of the cells due to observation geometry.

Local Perception & Planning Pipeline

Mathies et. al [Matthies et al., 2014] presented an approach to use disparity images generated by a stereo pair for obstacle avoidance. In this approach the occupied pixels in the disparity image obtained from the stereo pair are expanded to account for robot's size. The expanded disparity images are used as a spatial representation to plan collision free paths.

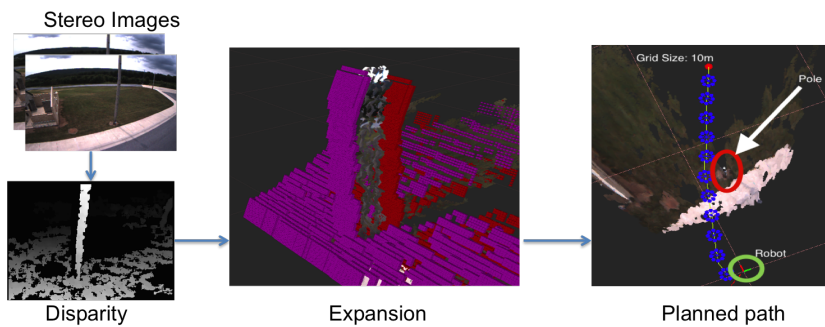


Figure 6.11: Planning pipeline based on inverse depth obstacle perception. The frontal expansion and back expansion are shown in pink and red point cloud around the original point cloud of pole. Planned path around the pole is also shown with the current robot position circled in green.

We maintain a similar pipeline to process the disparity images but improve the expansion step through the inclusion of the observation noise model in the disparity expansion. Furthermore, we compute two image expansions; frontal and back to probabilistically capture the occupancy region. Figure 6.11 and

Figure 6.13 show how the two images capture the pole obstacle. The frontal expansion is shown in pink point cloud and the back expansion in red. We also improve the path planning by using multiple disparity images to infer occupied volumes. The use of multiple disparity images allows the planner to reason about long range obstacles. The improved expansion algorithm and multi-image occupancy inference are presented in section 6.8. Furthermore, all the planned paths end at hover position with zero velocities ensuring safety of the vehicle. Figure 6.11 briefly shows the planning pipeline.

Local Perception

We use disparity image or inverse depth image for obstacle representation as it naturally captures spatial volume according to the sensor resolution [Gohl et al., 2015]. This representation is befitting for noisy stereo data as explained in Section 6.8. We employ C-space expansion where the original disparity image is expanded, allowing us to treat the robot as a point when doing collision checks during planning [Matthies et al., 2014].

Our method incorporates a stereo sensor error model and allows us to reason about space behind obstacles. We use an additional padding in disparity both in front and behind obstacles. This padding varies from 3σ for close obstacles to 1σ for far obstacles, where σ is the standard deviation of disparity error and the multiplier is represented by λ in later sections. By varying λ we ensure safe planning at short range and a more optimistic planning at long range. This enables the deliberative planning required for exploration tasks.

Disparity error and its effects

Disparity is a measure of the proximity of an obstacle. We can derive how close the obstacle is in depth using triangulation in stereo vision as follows.

$$z = \frac{bf}{d} \quad (6.11)$$

Where, z is the depth of a pixel(u, v) with disparity d , b is baseline and f is the focal length in pixels.

The actual 3D point can be derived as

$$P(x, y, z) = (uz/f, vz/f, z) \quad (6.12)$$

The accuracy of the stereo setup is drastically affected as the disparity decreases. The error in depth increases quadratically with depth as shown in equation(6.15). Differentiating equation(6.11) wrt d

$$\frac{\partial z}{\partial d} = -\frac{bf}{d^2} \quad (6.13)$$

$$\partial z = -\frac{z^2}{bf} \partial d \quad (6.14)$$

$$\partial z \sim z^2 \quad (6.15)$$

Disparity error is primarily caused due to correspondence error while matching pixels along the epipolar line. It can be modelled using a Gaussian pdf. Assuming correspondence error during disparity computation has a std deviation $\sigma = 0.5 \text{ pixels}$, we define the Gaussian pdf $\mathcal{N}(d, \sigma^2)$. Figure 6.12

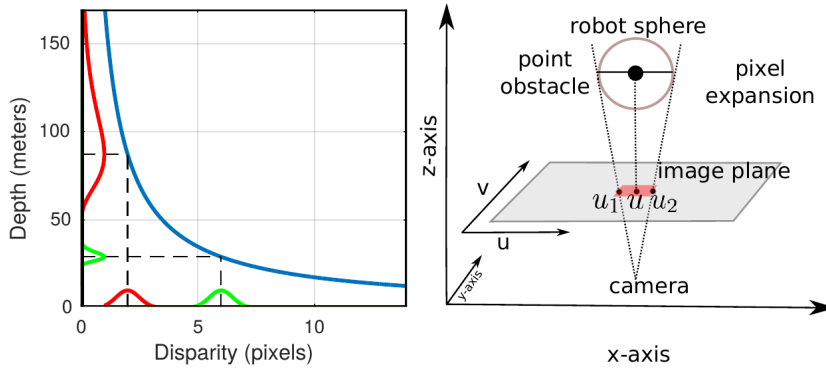


Figure 6.12: Left: Disparity vs Depth (blue) and probability distributions are shown in red and green. Red and Green PDF in disparity are same and easy to model but their corresponding Red and Green PDF in range vary and difficult to model. Hence we use inverse depth space to represent obstacles. Also, disparity i.e. inverse range captures space at multi-resolution suitable for registration of stereo sensor data. Right: Shows the pixel-wise expansion of a point obstacle according to robot size.

shows how this Gaussian pdf in disparity results in a difficult to model pdf for error in depth with an elongated tail on one side and a compressed tail on the other. This motivates to use disparity image space domain directly for occupancy inference rather than resorting to depth or 3D domain.

C-Space Expansion

C-Space expansion is required to represent obstacles such that a single point state query can be used for collision checks. Occupancy grids have been the default methods for registration of sensor data and C-Space expansion for occupancy inference. Usually point clouds are used to populate occupancy grids but point cloud generated using disparity images are highly uncertain at greater depths Figure 6.17(c) and hence occupancy grid based representation is infeasible. Moreover, 3D occupancy grids require a huge amount of memory to capture the planning workspace and hence fail to incorporate long range measurements available from stereo sensors. To overcome this limitation we use disparity images and apply disparity expansion step explained in section 6.8.

Disparity Expansion

In this section we explain the step of C-Space expansion as applied to disparity images. This step allows us to capture the volume occupied by an obstacle using two surfaces represented by two disparity images. These images represent front and back surface limits of the reported disparity. Each pixel in

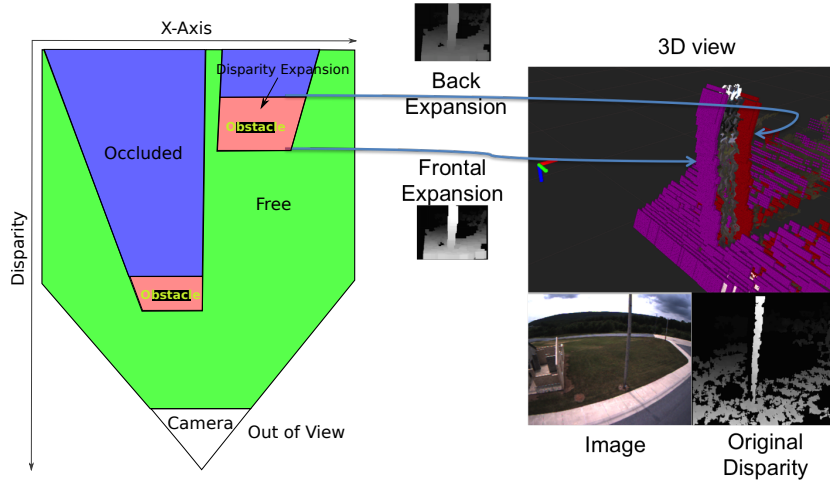


Figure 6.13: Disparity expansion shown as point cloud. The pink and red point cloud represent the foreground and background disparity limits.

these two images effectively captures the range of disparity based on robot size and the sensor error model as shown in the Figure 6.12. This process can be divided into two steps.

The first step expands disparities along the image XY axis Figure 6.12(right) i.e. an obstacle at some pixel (u, v) after inflation occupies a manifold of pixels from $[u_1, u_2]$ and $[v_1, v_2]$. This is achieved by traversing through the image row-wise first and then column-wise. This is similar to [Matthies et al., 2014] but we also incorporate sensor error. We omit the steps required to generate the look-up-table (LUT) to map $u \rightarrow [u_1, u_2]$ given disparity d and $v \rightarrow [v_1, v_2]$ given disparity d . Reader is advised to refer [Matthies et al., 2014] for generation of the LUT, but unlike looking up for the raw disparity value d from table we look up for $(d + \lambda\sigma)$, where λ is the sigma multiplier dependent on the range as discussed previously in Section 6.8.

The second step expands disparities to get new values for front and back images using equation(6.16). These images represent the maximum and minimum disparities for every pixel respectively.

$$\begin{aligned}
 z &= \frac{bf}{d} \\
 d_f &= \frac{bf}{z - r_v} + \lambda\sigma \\
 d_b &= \frac{bf}{z + r_v} - \lambda\sigma
 \end{aligned} \tag{6.16}$$

Where r_v is the expansion radius based on robot size, d_f and d_b are the computed front and back disparities which encompass the obstacle. As shown in illustration on left side of Figure 6.13, the red area around the original disparity of obstacle is the padding generated in the expansion step. This padding is based on the robot size and sensor error model.

Our approach uses the LUT as shown in Algorithm(12) which takes the

original disparity image D as input and processes it to generate the expanded frontal and back disparity images D_f and D_b respectively. The function $expand(d)$ implements equation(6.16) with $\lambda = 0$ and $r_v = 0$ to prevent double expansion in depth. The function $connectedComponent()$ searches for minimum disparity connected to the maximum disparity over steps of provided $range$ (set to a multiple of robot radius). This helps to find an obstacle bounding volume. We do not want to use the minimum disparity in a window as that can be located very far with no connection to the actual obstacle and hence the $connectedComponent()$ step is required.

Algorithm 12: Disparity Expansion Algorithm

```

Input: Disparity image  $D$ 
Output: Expanded disparity images:  $D_f, D_b$ 
for  $v = 1 : Height(D)$  do
  for  $u = 1 : Width(D)$  do
     $\hat{d} = ceil(D(u, v) + \lambda\sigma)$ 
     $[u_1, u_2] = LUT(u, \hat{d})$ 
     $V = D(u_1 : u_2, v)$  // Get vector of disparities
     $d_f = expand(max(V))$ 
     $d_b = expand(connectedComponent(d_f, range))$ 
    for  $i = u_1 : u_2$  do
       $D_f(i, v) = max(d_f, D_f(i, v))$   $D_b(i, v) = min(d_b, D_b(i, v))$ 
    end
  end
end
  
```

Algorithm(12) does row-wise expansion and its result is then subject to column-wise expansion in a similar fashion with λ and r_v set to default values when using $expand()$ function. The expanded disparity images constitute a single snapshot volumes occupied by obstacles. To maintain a spatial memory we create a pose graph consisting of multiple expanded disparity images as described in the following section.

Pose Graph of Disparity Images

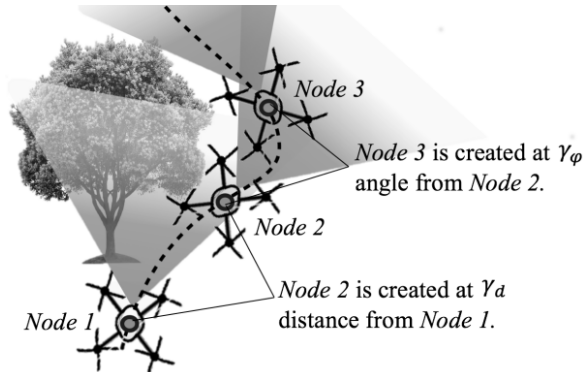


Figure 6.14: Pose Graph of expanded disparity images. Dashed path shows robot motion and stored nodes in the graph are shown as triangles. Nodes are stored at intervals of distance and orientation.

The motivation to maintain spatial memory of the previously seen environment as the vehicle is moving using a pose graph is because of the following

reasons:

1. Previously seen obstacles might not be visible in the current image.
 - (a) The stereo sensor has a minimum range dependent on maximum perceivable disparity.
 - (b) Obstacles get occluded in different views.
 - (c) The field of view is limited.
2. Maintain a pose graph of disparity images (measurements) with nodes at regular intervals of distances and angles as shown in Figure 6.14.
3. Allows occupancy inference using multiple measurements.

Algorithm(13) shows how we construct this graph. Each node in the

Algorithm 13: Pose Graph Algorithm

```

Input:  $D_f, D_b, Pose, N_{graph}, \gamma_d, \gamma_\psi$ 
Output: Pose Graph of Expanded disparity images: Graph
 $T_s^w \leftarrow Pose$ 
Node = createNode( $T_s^w, D_f, D_b$ )
if Graph.size() == 0 then
  | Graph.push_front(Node)
  | Graph.push_back(Node)
end
PrevNode = Graph.begin()
pos_err = distance(Pose, PrevNode)
ang_err = angle(Pose, PrevNode)
if pos_err >=  $\gamma_d$  || ang_err >=  $\gamma_\psi$  then
  | if Graph.size() ==  $N_{graph}$  then
  | | Graph.pop_back()
  | end
  | Graph.push_front(Node)
end
Graph.pop_back()
Graph.push_back(Node)

```

graph is comprised of the following:

1. D_f
2. D_b
3. T_s^w which is the transform between the processed sensor measurement(D_f, D_b) and world frame.

The algorithm takes as input the current robot position $Pose$, processed disparity images D_f, D_b , maximum number of nodes N_{graph} and two tolerance parameters γ_d, γ_ψ for position and angular displacement respectively. The constructed graph is used to project a given world point into all node images and do occupancy inference. Occupancy inference using the set of disparity images in the graph is explained in subsequent section.

Occupancy Inference

Evidence grids or occupancy maps are methods to allow fusion of different measurements taken over time. By maintaining a pose graph of expanded disparity images, we can also take advantage of similar fusion without building an occupancy grid which are not suited for stereo data as discussed previously. We devised an occupancy inference method by fusing information from all the images in the graph using the stereo sensor error model. Given the standard deviation of correspondence error σ , we compute confidence of a disparity state in the following manner.

$$C(d) = \frac{(d - \sigma)}{d} \quad (6.17)$$

Confidence measure from equation(6.17) gives us a measure of how much can we trust a given disparity for occupancy inference. Thus, long range or low disparity, uncertain measurements have low confidence and update the occupancy with lower values. We further discount measurements that mark an area safe or potentially safe(occluded) by 0.5 to be more conservative about clearing areas previously marked occupied. It should be noted that the potentially safe areas are behind obstacles and have lower disparity state, hence their contribution to occupancy clearance is less due to lower confidence value. In our experiments we get the final occupancy measure by projecting a world point P using equation(6.18) and equation(6.12) in disparity images of all nodes in the graph and accumulating the occupancy cost according to Table(6.4):

Check	Remark	occupancy cost $occ(d_s)$
$d_s > d_f(u, v)$	safe	$-0.5C(d_s)$
$d_s < d_f(u, v)$ and $d_s > d_b(u, v)$	obstacle	$C(d_s)$
$d_s < d_b(u, v)$	potentially safe	$-0.5C(d_s)$

Table 6.4: Occupancy update

Collision Checking

Collision checking is used to plan a new path and to validate if an existing path is safe to follow. Collision checking is performed using the following mapping of a 3D world point P to image pixel I with disparity d_s :

$$P(x, y, z) \leftrightarrow I(u, v, d_s) \quad (6.18)$$

A state is in collision if the occupancy measure as shown in equation(6.19) crosses a pre-defined threshold γ .

$$Occupancy = \sum_{nodes} occ(d_s) \quad (6.19)$$

$$Occupancy \geq 0.0 \quad (6.20)$$

If the occupancy for a state is below the threshold, we consider that state as not occupied by an obstacle.

Planning

We use a sampling based planner, BIT* [Gammell et al., 2014] to draw samples in 3D space which are checked for collision as described in 6.8. The output is a collision free path connecting start to goal state.

In our experiments we found that disparity images fluctuate around obstacle edges leading to unwanted replanning due to the current plan being in collision. To remedy this we used two threshold values. A lower value γ_{low} is used during planning to find a path i.e. obstacles are observed sooner even at long distances and hence a more conservative path is obtained. A higher threshold value γ_{high} is used to check the current plan for collision and do replanning in case of collision. The advantage of using two threshold values is that an initial plan is found using a more conservative occupancy map while the replanning is done using a more reliable occupancy map. The reliable occupancy map is not affected by fluctuations in the disparity maps. The thresholds are chosen such that collisions at close range are always detected but have great advantage to not force replanning due to less reliable and fluctuating observations at long range when planning paths to longer distances. In our experiments we have planned paths at distances longer than 100m (Figure 6.17). Figure 6.11 shows a planned path that avoids a pole obstacle. This path is sent to the motion controller of the vehicle.

Motion Control

We developed a path tracker similar to [Hoffmann et al., 2008]. It takes the current trajectory and uses feed-forward velocities specified in the trajectory and generates final velocity and heading rates for the low level velocity controller. The low level velocity controller runs on the quadrotor's flight control unit.

The trajectory controller obeys the dynamic limits of the robot and limits the velocities in event of hard turns or sharp changes in trajectory. This allows the planner to generate simple waypoint based paths and rely on the trajectory control to obey vehicle dynamics.

Local Planning Experiments

We conducted most of the experiments in the highlighted area shown in Figure 6.15. Some of the features of region were narrow trails, dense foliage and varying height tree line, all of which made for challenging and interesting obstacles. Tests involved manual take-off and sending a list of sparse global waypoints to the obstacle avoidance system with the desired velocity. Sparse global waypoints allowed obstacle avoidance system to plan around obstacles



Figure 6.15: Location where experiments were carried out: highlighted area

determining vehicles path and safety. Table(6.5) lists the values we used for conducting the experiments.

Parameter	Value
Baseline: b	0.35m
Focal length: f	514.17 pixels
Correspondence error: σ	0.5
Connected component range: $range$	$2r_v$
Robot radius: r_v	1.5m
Lenient Occupancy Threshold: γ_{high}	1.8
Strict Occupancy Threshold: γ_{low}	0.9
No. of nodes in Pose graph: N_{graph}	10
Displacement between nodes: γ_d	1.5m
Angle between nodes: γ_ψ	30°

Table 6.5: Parameters Used

Local Planning Results

Figure 6.16 shows the time taken to process a single disparity image to compute the frontal and back expansions using Algorithm(12) on the on-board ARM computer. In our experiments we used CPU version at 320×240 resolution because the GPU was used for semantic classification algorithm as concurrent part of the experiments. A pose graph using Algorithm(13) was created and used for collision checks using equation(6.19). Using our approach a single occupancy inference and collision check takes on average $0.01ms$. Given $100ms$ between each frame we can do about 2000 collision checks which was usually sufficient for the BIT* planning algorithm.

Figure 6.17(a) Shows planned path going through two low height trees. The top left is the disparity image with left camera image shown on top right.

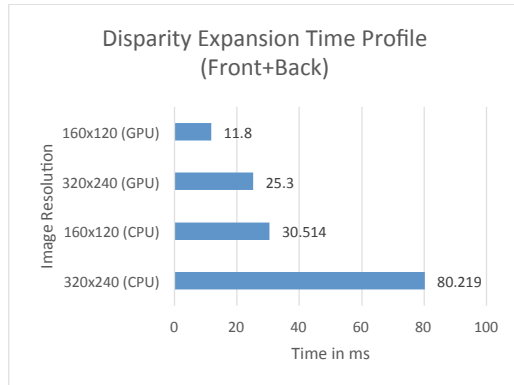


Figure 6.16: Time profile of expansion step.

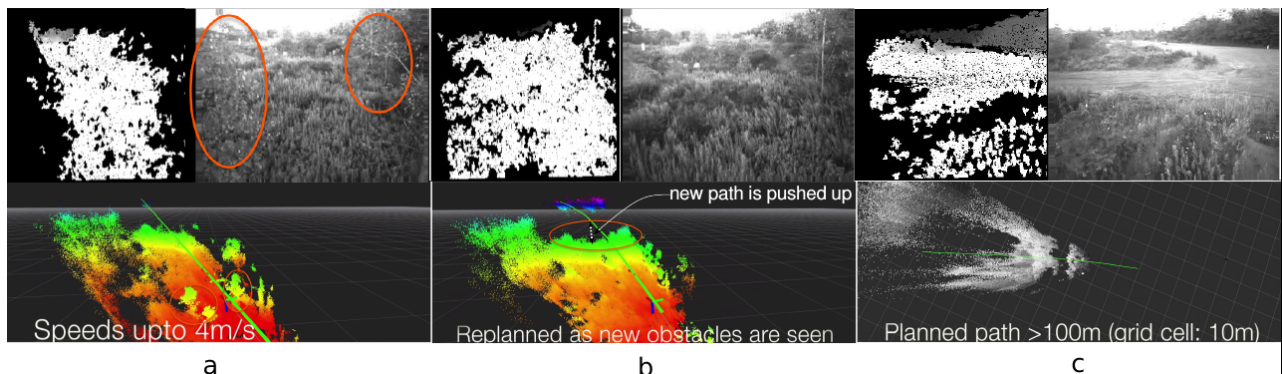


Figure 6.17: Point cloud is shown at the bottom in all three figures for reference. Point cloud is colored by height in (a) & (b) and by actual intensity in (c). (a) Planned path(green) between low trees highlighted in red ellipses (b) Replanned(green path) as more observations are made, marked in red ellipse, (c) Long range planning horizon. The point cloud shows the noisy measurement but even noisy information allows to infer occupancy at long distances.

The point cloud is only for visualization purpose and the trees are marked in red ellipses. Although the trees are not completely visible in the current disparity image, they are still a part of obstacles as they were seen at previous robot positions and hence stored in the pose graph. Without the pose graph these trees would have been invisible to the robot. Thus the pose graph helps in keeping memory of obstacles which were seen previously but can't be observed as they exceed the limit of maximum possible disparity after robot motion.

Figure 6.17(b) shows the previous path was re-planned and pushed up as more observations of the bushes/trees are made at long range are marked as obstacles at approximately $30m$ distance from the robot. This was possible due to fusion of occupancy using several disparity images in the pose graph.

Figure 6.17(c) emphasises the advantage of planning in disparity space at long distances. At greater distances the point cloud is very noisy but we are able to get some information about occupancy by using all the sensor data. While occupancy grids would have huge impact, both memory wise and computationally to use all this data, our approach is able to incorporate all the information using minimalistic image space representation and do better occupancy inference.

Figure 6.18 shows the reactive nature of our approach. For this experiment the robot was allowed to find a plan outside the sensor's field of view and was given a goal point in right direction. As the robot follows the plan and turns right, an obstacle obstructing its path is detected and a new plan avoiding it is generated. This happened at a speed of $4m/s$ hence implying our approach quickly reacts to newly seen obstacles.

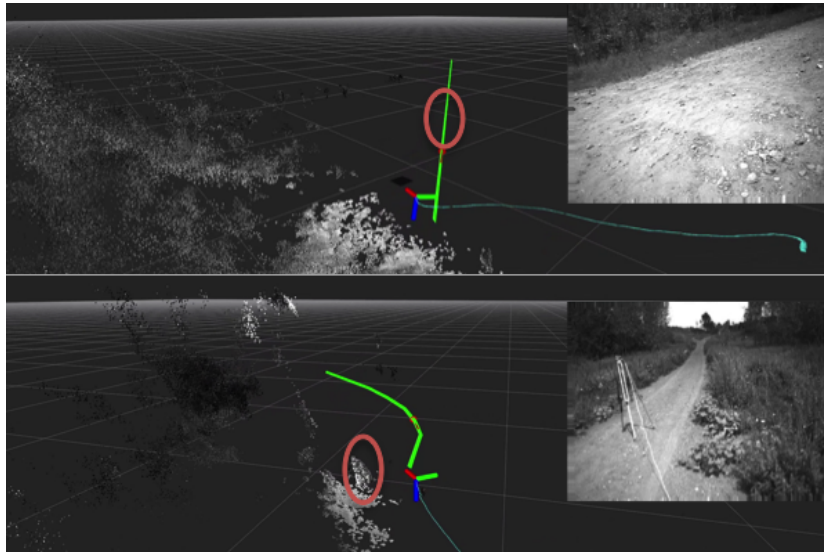


Figure 6.18: Reactive Planning at $4m/s$: Top image shows the robot has planned to go right with unseen obstacle marked in red ellipse. Bottom image: after banking right an obstacle obstructs the previous plan and a new plan avoiding it is generated.

Using the parameters specified in Table 6.5, if the robot moves $15m$ main-

taining 10 nodes and assuming a maximum of only $100m$ depth ($1.79pixel$ disparity) per image our approach uses approximately 38% of the memory required by a gridmap of cell size $1m^3$ covering the same volume. This is the case when using a gridmap of large cell size meaning a very coarse resolution. For a better resolution gridmap will require even more memory.

More than 100 runs were executed with approximately 1.6 hours in autonomous mode, covering a cumulative distance of approximately $1.5Km$. The maximum speed was capped at $4m/s$. Our approach allowed us to plan to distances greater than $100m$ as shown in Figure 6.17(c). Average distance to goal was $36m$. The standard deviation of length of planned paths from straight line paths was on average of $1.38m$ with a maximum of $30m$. This shows that in most cases planned paths were close to a straight path but with slight deviation to avoid obstacles.

7

Conclusion

The central thesis of this document is that -

For mobile robots operating in partially known environments, active gathering data and concurrently reasoning about motion constraints produces higher rates of information gain and higher speeds of safe operation.

Mobile data gathering agents have an implicit need to gather data to navigate the environment safely, while there is an explicit need to gather task-specific data. Both explicit and implicit needs for gathering data have to contend with the physical constraints of the agent like sensing bandwidth, dynamic, and energy constraints. The work presented in this document has focussed on enabling safe, efficient data gathering agents in physical spaces.

In section 7.1 we summarize the critical results presented in this document and present the significance of contributions in section 7.2. We discuss the theoretical structures uncovered during this work in section 7.3 and limitations and open research questions in section 7.4.

7.1 Results Summary

We demonstrated that to ensure safety, near-optimal utilization of dynamic constraints through emergency maneuver library (*EML*) leads to faster, safer operations with reduced sensor range requirements as compared to methods that approximate vehicle dynamics. Using survivability as a metric to generate *EML* enables full utilization of vehicle dynamics for safety. Survivability is monotonic, sub-modular set function, enabling the generation of near-optimal trajectory sets using greedy algorithms for an NP-hard problem. Using *EML* on rotorcraft enabled them to fly at 1.8 times the speed (56 m/s) as compared to techniques that approximate motion (dynamic) constraints of the vehicle.

EML combined with policy-based sensor planning provides a framework to address the need for implicit data gathering for the safety of mobile robots. Modeling vehicle dynamics for safety enabled us to define the pertinent region from which information about obstacles needs to be gathered. We showed that this pertinent volume definition could then be used to learn an

optimal policy for active data gathering. This optimal policy also provides worst-case performance guaranty. We developed an algorithm to improve these worst-case guarantees online to account for occlusion and known space. Using the same worst case principle, we developed a policy for the explicit data gathering need of evaluating landing zones. The augmented online policy enabled maximum safe speed of 60 m/s for rotorcraft which is 25% more than the worst case policy and 300% more than data gathering schemes that fail to account for agent’s dynamics constraints.

Motion planning for explicit data gathering in physical spaces needs to take the cost of moving into account in the presence of physical (path-length) constraints. Data gathering agents are often deployed to gather information with a minimum resolution requirement. For systems capable of gathering data at multiple resolutions, it is often the case that the actions that lead to low-resolution information do not lead to direct rewards. We proved that to generate efficient solutions to multi-resolution data gathering problem while accounting for path-integral constraints like total route length agents need to reason about belief space dynamics. Guided by this result, we presented an information-theoretic formulation of budgeted data gathering and develop *RAOr-G*, an anytime algorithm for budgeted, multi-resolution data gathering.

The critical insight used to overcome the dimensionality of the problem is that the order of visiting the viewpoints is independent of the reward gained by visiting those viewpoints. This structure enabled us to take guided random walks in the space of near-optimal routes to construct near-optimal IPP solutions incrementally. *RAOr-G* produced solution 2 orders of magnitude faster than other solutions that provide optimality guarantees and 1.75 times better solutions other sampling-based algorithms. More significantly, the multi-resolution information gain paradigm, where the data gathering agent focuses its sensing bandwidth and limited operation time on pertinent information result in coverage of areas 7 times larger than a non-adaptive paradigm that fails to model the physical capability of the agent to gain information at multiple resolutions. In the next section we present the significance of the algorithms that enabled these improvements with respect to the state of the art.

7.2 Significance of Contributions

Active data gathering is a design/control paradigm where the data being gathered during experimentation is optimized to meet the objectives of the experiment. The desire is to meet experiment’s objectives in minimum cost or to maximize a reward function. In digital spaces, where the cost of gathering data is independent of data points gathered, the problem is well studied and even though its NP-Hard, near-optimal polynomial time solvers exist.

Mobile data gathering agents are bound by their dynamic, sensory and battery limitations. Active data gathering in digital spaces does not have to

contend with these motion constraints. Reasoning about motion constraints exponentially increase the search space. Majority of the state of the art methods address this increase in search space through following three approaches: 1. Algorithms that ignore motion constraints and use heuristic driven policies or Pareto-optimal policies, such methods do not provide any performance guarantees and are not scalable. 2. Algorithms that solve for routes that model motion constraints offline. Although such methods provide guarantees, however, their runtimes do not enable online generation of solutions. 3. Decision-theoretic POMDP solvers have been able to address the active data gathering problem for localization successfully but the dimensionality of active data gathering for safety or mapping makes them unsuitable to run online.

We have focused on developing safe, mobile data gathering agents while modeling their motion constraints. We have demonstrated significant performance increase in agent's data gathering performance as a result of motion constraint modeling and in the process developed a framework that enables safe, efficient physical data gathering with performance guarantees. The techniques developed work in partially known environments and ignore the pose uncertainty of the agents themselves. Our main algorithmic and theoretical contributions are -

- Describing an algorithm for guaranteeing the safety of a mobile agent operating in a partially known environment while fully utilizing its dynamics. The algorithm is independent of the planning pipeline and guarantees upper-bound on the run-time, while making no assumptions about the dynamics of the vehicle.
- Proving the equivalence of optimizing sensor trajectory to minimize vehicle trajectory cost and optimizing the sensor trajectory to optimize an information theoretic reward function. Using the equivalence and eml based safety to define a near-optimal algorithm for active data gathering for the safety of mobile agents in partially known environments.
- Demonstrating that near-optimal solvers for budgeted, multi-resolution information gathering need to model high-dimensional belief space dynamics.
- Describing an algorithm to solve budgeted, multi-resolution data gathering problems online by exploiting the structure that the reward for visiting viewpoints is independent of the order in which they are visited.

Through deployment of these algorithms over multiple platforms and testing over hundreds of hours of operations, we have shown that active data gathering while reasoning about motion constraints leads to a significant gain in performance of autonomous mobile robots operating safely in partially known environments. In the next section, we discuss some of the underlying

mathematical structures that we have discovered while developing active gathering techniques that can reason about motion constraints. In section 7.4 we present limitations of the current framework and possible future directions, before presenting final thoughts in section 7.5.

7.3 Discussion

In this section, we discuss the underlying mathematical structures we have uncovered while developing solutions for planning for data gathering and safety - 1. Survivability model for safety, 2. Equivalence of trajectory cost minimization and contextual information gain maximization for sensor motion planning, 3. Presence of informative actions in multi-resolution information gathering, 4. Independence of reward for a set of viewpoints from the order in which they are visited. We highlight how we utilized these while discussing their broader applicability. We then shift our focus towards the limitations of the presented algorithms and suggest future direction of work.

Survivability Model for Safety

In chapter 3, we used the insight that the safety of a vehicle can be guaranteed by using an offline computed library. Previous works on the safety of autonomous robots can be broadly divided into two paradigms. One of the paradigms is to make sure that the vehicle can stop within the sensor range while applying maximum allowed longitudinal deceleration [Scherer et al., 2012c, Goerzen and Whalley, 2011, Adolf and Dittrich, 2012]. The stopping distance based velocity limit does not exploit the full dynamics of the vehicle, leading to conservative velocity limits.

Another paradigm is to simplify the non-linear dynamics of the UAVs and plan a path that is guaranteed to stay within the known unoccupied region. Mixed integer linear programming is used in [Schouwenaars et al., 2004] to plan paths that stay within the known region. Simplified dynamics in a sampling-based graph is used in [Frazzoli et al., 2002] while limiting the maximum planning time to ensure safety. The assumption is that the planner can always plan an obstacle-free path if allowed to run until the maximum planning time. [Enright et al.] uses Dubins curves to plan paths within the known space. These methods also suffer from not being able to exploit the vehicles full dynamic capabilities.

Use of an offline computed library enables full utilization of the dynamics of the vehicle to keep it safe. We proved that safety could be modeled as survivability. This critical underlying structure enables the use of the sub-modular and monotonic nature of survivability and develop a generic near-optimal algorithm that can ensure a vehicle's safety irrespective of the complexity of its dynamics while making efficient utilization of free space.

As a result, the algorithm could guarantee the safety of rotorcraft at speeds of up to 60 m/s, while stopping distance based methods failed at speeds over 40 m/s. Modeling safety as survivability also helped define a pertinent volume from which obstacle data needs to be gathered to keep the vehicle safe.

A drawback of the emergency maneuver library is that it is fixed and does not change with the environment. In geometrically dense scenarios, it is possible that a fixed sparse emergency maneuver library will lead to a conservative performance of the vehicle. However, the sub-modular and monotonic nature of survivability enables construction of adaptive sets of libraries and overcome this drawback, as shown by [Dey et al., 2011]. Hence the underlying structure of modeling safety as survivability has not only led to 1.5 times the maximum safe speed increment in the application covered in the thesis but also enables an extension of the idea to adaptive settings without any assumptions on vehicle dynamics.

Trajectory Cost Minimization & Information Gain Maximization for Sensor Motion Planning

In chapter 4, we proved that the optimizing sensor trajectory to minimize the vehicle trajectory cost, given a vehicle path is equivalent to optimizing the sensor trajectory to maximize contextual important information gain. This property enables projecting of the reward of sensory actions in an information space. The equivalence property combined with the use of emergency maneuver library for safety enabled us to define a worst case scenario for a sensor, gathering data to ensure vehicle's safety. We then exploited symmetries in the problem to learn an optimal policy offline for the said worst case scenario and augment the policy performance online using sparse features. The basic equivalence structure helped define an active sensor motion planning algorithm that can escape the curse of dimensionality as faced by POMDP based solvers [Myers and Williams, 2010], while still providing performance guarantees. We also used the same equivalency structure to generate an information theoretic reward function for the explicit need for data gathering for LZs as well. The sensor motion planning method developed enabled three-fold increase in maximum safe speed over passive sensing, and an approximate six-fold increase in effective sensing range.

However, the equivalence property does not address how to derive the information theoretic reward function from the original trajectory optimization problem, in fact, we prove that deriving the reward function is as hard as solving the original problem. One way to address this problem is to move away from model-based methods, to model free methods for inference of reward function. However, the dimensionality of the problem prohibits the direct application of such methods.

Presence of Informative Actions in Data Gathering

In chapter 5, we demonstrated that the problem of multi-resolution data gathering using UAVs has informative actions in its action space. The action to gain height to gather low-resolution, large-scale information. We show that the presence of such action makes it imperative for the solver to model belief space dynamics. Rendering MDP based approximate POMDP solvers, [Little et al., 2007, Yoon et al., 2008, Littman et al., 1995a] and related imitation learning based methods ineffective. This property of multi-resolution data gathering motivated the information theoretic approach we adopted to model the problem, chapter 6. We do not address the issue of automatic detection of informative actions. However one can use MDP solvers to detect their presence and decide the suitability MDP based approximate POMDP solvers and related imitation learning-based methods for an application.

Independence of Reward Function For Explicit Data Gathering from Order of Viewpoints

In chapter 6, we leverage the fact that for data gathering applications in static environments and with no uncertainty about the position of the robot, the reward is a set function of viewpoints and is independent of the order in which those viewpoints are visited. This property of the reward function enables us to sequentially solve the set selection and the order problem. The sequential break down of the problem results in effective exponential search space reduction, leading to the development of *RAOr*, an online solver for budgeted, multi-resolution informative path planning (IPP) problem. *RAOr* has enabled us to solve IPP problem onboard systems with limited computational power.

Together these approaches have resulted in hundreds of safe flight hours, enabling data gathering systems to fly at 3 times the speed and to gather data of areas 7 times larger than possible from state of the art,

7.4 Future Work

Combining Implicit and Explicit Data Gathering

Emergency Maneuver Library (*EML*) enables the vehicle to stay safe while operating in partially known environments. We have used the emergency maneuver library with a myopic one-step look ahead, independent of path planning. Treating path planning and safety independently enables real-time guaranteed safe planning for systems with highly non-linear dynamics, figure 7.1. But this independence assumption potentially leads to sub-optimal system behavior.

For example, in figure 7.2 if the planner and *EML* are operated independently, the planner maximizes the distance from obstacles and finds a path

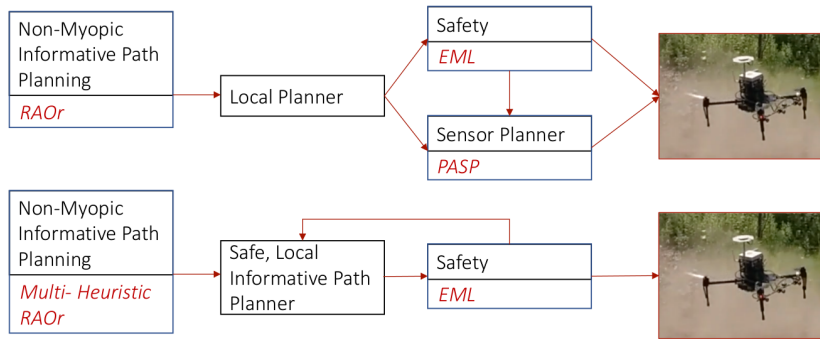


Figure 7.1: Top: Current data gathering pipeline. The implicit and explicit data gathering is considered separately, leading to undesired behavior like slower navigation and, stopping for safety. **Bottom:** Combined data gathering pipeline with multi-heuristic *RAOr* for faster budgeted informative path planning, and safe, local informative path planner, that reasons about trade-off between implicit and explicit data gathering overcoming drawbacks of the current approach.

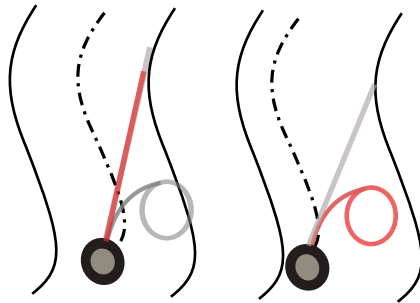


Figure 7.2: Left: Motion planner operates independently of the safety library. Resulting in maximizing the distance from walls of the corridor and slower navigation speeds. **Right:** If the motion planner is cognizant of *EML* and safety constraints, it can identify a faster solution to fly through the corridor while being close to a wall.

in the middle of the corridor, leading to a slower operation of the vehicle. Whereas if the planner actively optimizes the maximum safe speed while cognizant of the emergency maneuver library takes a path much closer to one of the walls of the corridor, enabling faster safe navigation.

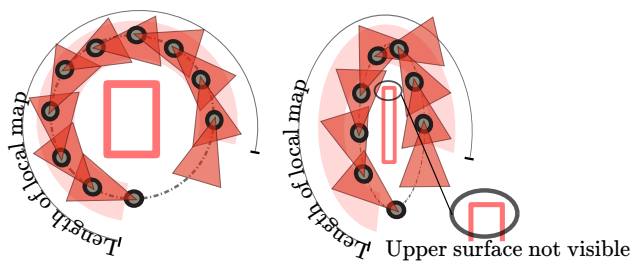


Figure 7.3: Left: Vehicle flies a constant speed around the target to gather its data, while myopically looking forward as and when required. **Right:** The same approach does not work in this scenario, as there is a sharp turn in the path, leading to the vehicle missing essential information about the target as it ensures safety.

Efficient data gathering also requires the vehicle to gather relevant data while being safe. The simplest approach for solving this problem is to gather data for safety when required, while focusing sensing bandwidth on explicit data gathering needs as the vehicle flies at a constant velocity. Figure 7.3, explains a scenario where myopic constant velocity strategy works for a circular path, but the same approach leads to missing pertinent information if the curvature of the path is relatively higher.

The related informative path planning problem can be posed as a POMDP. However, solving such a formulation is intractable. Bircher et al. [Bircher

et al., 2016], tackled this high dimensional problem by embedding information theoretic reward function in randomized sampling based planning techniques. The suggested approach works well for short-horizon motion planning problems but the planner is too computationally intensive for longer horizon problems, moreover it provides no performance guarantees. In the next section, we identify how a modified version of *RAOr* can be used to address the problem.

Safe Local Informative Path Planning

In chapter 6, we exploited the structure in the reward function that the order in which viewpoints are visited is independent of the information or reward gained by visiting the viewpoints. The order of visiting the viewpoints only affects the cost. We developed *RAOr* using this property of the reward function. Adding safety as a constraint to the optimization of the reward function, does not change its property of being order independent. Hence *RAOr* algorithm is still applicable for solving safe informative path planning problems.

However, it cannot be used directly, since to run the *TSP* in the *RAOr* algorithm, we need to find a dense graph with costs associated with each of the edges. We want to collect maximum information in minimum amount of time, hence the cost of each edge is the time taken to travel that edge by the agent while being guaranteed safe. However, the addition of safety constraint leads to the dependence of time of travelling between viewpoints on the complete motion history of the vehicle. The speeds at which the vehicle can operate at, under the safety constraint is dependent on known free space and the free space observed by the vehicle is dependent on the path it has taken through the environment. This dependence of time of travelling on the vehicle motion history exponentially increases the search space. Moreover, to find an optimal cost, the optimizer needs to forward simulate the observations to determine the speeds at which the vehicle can fly at, which is computationally expensive.

Hence, to enable the use of *RAOr* for safe informative path planning, we have to define an optimizer that can overcome these challenges to efficiently find a near-optimal time trajectory. To develop an online optimizer, we first ignore the motion history of the vehicle. Now, we need to find an optimal connection between two vertices, x_s and x_e , where each vertex specifies a maximum speed, position in $3D$, and desired heading. The number of paths between these vertices is $n_a^{n_s}$, where n_a is the number of discrete accelerations within the dynamic limits of the vehicle and n_s is the number of discrete waypoints between x_s and x_e . It is infeasible to forward simulate exponential ($n_a^{n_s}$) number of paths while forward simulating the corresponding observations received to find the optimal route.

We use the insight that if the vehicle's heading is fixed and the known free

space observed while the vehicle follows a trajectory is independent of its speed, the problem of connecting x_s and x_e with an optimal safe trajectory is reduced to a bang-bang controller, with the complexity that is linear in n_s . We also assume that the fixed heading that enables fastest safe trajectory from x_s to x_e is the one that aligns with the path.

Using the insight and assumption above, we divide the problem of finding a near-optimal connection between x_s and x_e into three parts, with bang-bang controller being the optimal controller for each segment. We split the problem into finding the distance along the path (s_{tp0}) and the speed (v_{tp0}), where the vehicle aligns with the direction of travel, and finding the distance along the path (s_{tp1}) and the speed (v_{tp1}) where the vehicle starts turning to align with $x_e.\psi$. This reduces the problem of finding a trajectory from x_s to x_e into finding a trajectory between x_s to s_{tp0} , s_{tp0} to s_{tp1} , and from s_{tp1} to x_e . Each of these problems can be solved by using linear time bang bang controllers. This separation of the problem, leads to a reduction of an exponential ($n_a^{n_s}$) search space to a polynomial $n_a^3 n_v^3$ search space. Also, the number of observation simulations required reduce from $n_s n_a^{n_s}$ to n_s , if we ignore the known space observed by the vehicle while it turning.

The above described splitting of the connect problem leads to the Algorithm 15. In the following we describe the algorithm after defining some primitives and notation. Let, $G = [V, E]$ be the graph of viewpoints, with V representing the vertices and E the edges. $x_i \in V$ is a vertex, $x_s \in V$ and $x_e \in V$ are starting and ending vertices, b_0 is the initial belief of the robot, ψ_{path} is the heading of the path, and $\dot{\psi}$ is the maximum allowed heading rate of the vehicle, r_e is the effective range of the vehicle, v_{def} is the default speed required at the viewpoints, and a_{max} is acceleration limit of the vehicle. Primitives are defined as follows -

GetMaxTP₀(.) - Returns the maximum distance along the path at which the vehicle has to turn to align itself to the direction of the path. It is the maximum distance along the path for which the vehicle can be considered safe without adding any observations to b_0 .

GetMinTP₁(.) - Returns the minimum distance along the path at which the vehicle can to start turning to align itself to $x_e.\psi$. It is the minimum distance along the path from which x_e can be observed to be in known free space.

CacheRepresentation(.) - Returns a cached, indexed set of observations, $\mathcal{B}_{0:s}$ where s is the path length, along the path starting at x_s and ending at x_e , while assuming that the heading of the vehicle is aligned with path.

RepUseList(.) - Returns a cached, indexed set of observations, $\mathcal{B}_{s_0:s_1} \in \mathcal{B}_{0:s}$, that can be used from the cached set of observations, $\mathcal{B}_{0:s}$, given s_0 and s_1 are s_{tp0} and s_{tp1} respectively.

VSafe(.) - returns an ordered set of maximum safe speeds given b_0 and $\mathcal{B}_{s_0:s_1} \in \mathcal{B}_{0:s}$ and an *EML*.

SafeConnect(.) - returns the minimum time safe trajectory from x_s to x_e by

searching through all possible s_{tp0}, v_{tp0} and s_{tp1}, v_{tp1} , while respecting the upper velocity constraint provided by V_{safe} .

MinTimeBangBangController(.) - returns a minimum time trajectory between two points, while respecting V_{safe} as the upper bound speeds and the minimum time required to turn from the starting vertex to some end point and velocity.

MaxTimeBangBangController(.) - returns a maximum time trajectory between two points, while respecting V_{safe} as the upper bound speeds and the minimum time required to turn from the starting vertex to some end point and velocity.

ComputeSafeDMatrix(.) - returns the minimum safe time matrix D , that acts as the cost matrix for the *TSP* in Algorithm 18.

OptimizeRoute(.) - optimizes the speed and heading profile along a route while modeling the dependence of safe speed of the robot on a given path segment on the history of motion the robot. The routine uses algorithm 14 and sensor planning as defined in chapter 4 as sub-routines.

Algorithm 18 is similar to Algorithm 9 but can be used for solving the combined safe, local, informative path planning problem. The only two differences being, it uses the D_{safe} matrix instead of the D matrix for cost of the edges and uses the *OptimizeRoute(.)* function to optimize the best route once it has been found.

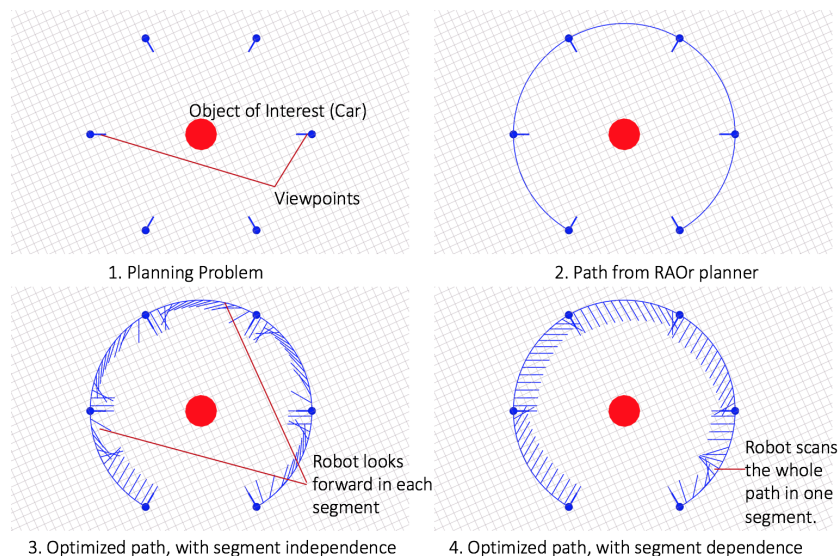


Figure 7.4: 1: In this scenario, the viewpoints are $10m$ away from the target, the vehicle is equipped with a sensor with a range of $22m$ and field of view of 60 degrees, if a viewpoint views a unique sector of the cylinder, $+1$ reward is gained. The maximum speed of the vehicle is restricted to $3 m/s$. **2:** An informative route planned by using *RAOr*. **3:** A safe informative route planned by using *SafeRAOr* using path independence assumption. **4:** A safe informative route planned by using *SafeRAOr* while modeling path dependence. Given a maximum heading rate of the vehicle, $\dot{\psi} = 1.06 \text{ rad/s}$, the trajectory generated by assuming paths are independent leads to total time taken by $38.24s$, while the trajectory generated by modeling path dependence takes $34.37s$ to cover the same viewpoints. It takes the planner $1.12s$ to generate the trajectory.

Although we did deploy *SafeRAOr-G* on a vehicle operating in the field, it was only tested for one scenario, figure 7.4. There are open research questions that we need to answer in order to prove the effectiveness of this planner.

1. *Under what sensor and environment characteristics is the assumption of*

Algorithm 14: SafeConnect

Input: $x_s, x_e, s_{tp0}, s_{tp1}, \mathcal{V}_{safe}, \psi_{path}, \dot{\psi}, a_{max}, v_{def}$
Output: Minimum time safe trajectory from x_s to x_e with s_{tp0} & s_{tp1} as turning points
 $\sigma_{min} = MAXDOUBLE$
for $v_{tp0} = v_{def} : -v_{res} : 0$ **do**
 $[valid, \sigma_{tp0}] = MinTimeBangBangController(x_s, s_{tp0}, v_{tp0}, \dot{\psi}, \psi_{path}, a_{max}, \mathcal{V}_{safe})$
 if $\neg valid$ **then**
 $[valid, \sigma_{tp0}] =$
 $MaxTimeBangBangController(x_s, s_{tp0}, v_{tp0}, \dot{\psi}, \psi_{path}, a_{max}, \mathcal{V}_{safe})$
 end
 if $\neg valid$ **then**
 continue
 end
 for $v_{tp1} = v_{def} : -v_{res} : 0$ **do**
 $[valid, \sigma_{tp1}] =$
 $MinTimeBangBangController(s_{tp0}, v_{tp0}, s_{tp1}, v_{tp1}, \dot{\psi}, \psi_{path}, a_{max}, \mathcal{V}_{safe})$
 if $\neg valid$ **continue**
 for $v_f = v_{def} : -v_{res} : 0$ **do**
 $[valid, \sigma_f] =$
 $MinTimeBangBangController(s_{tp1}, v_{tp1}, x_e, v_f, \dot{\psi}, \psi_{path}, a_{max}, \mathcal{V}_{safe})$
 if $\neg valid$ **then**
 $[valid, \sigma_f] =$
 $MaxTimeBangBangController(s_{tp1}, v_{tp1}, x_e, v_f, \dot{\psi}, \psi_{path}, a_{max}, \mathcal{V}_{safe})$
 end
 if $\neg valid$ **then**
 continue
 end
 if $Time(\sigma_{tp0}) + Time(\sigma_{tp1}) + Time(\sigma_f) < Time(\sigma_{min})$ **then**
 $\sigma_{min} = \sigma_{tp0} + \sigma_{tp1} + \sigma_f$
 end
 end
 end
end
return σ_{min}

Algorithm 15: OptSafeConnect

Input: $G = [V, E], x_s, x_e, \psi_{path}, \dot{\psi}, a_{max}, v_{def}, b_0, r_e$
Output: Minimum time safe trajectory from x_s to x_e
 $mintime = MAXDOUBLE$
 $s_{tp0}^{min} = 0$
 $s_{tp1}^{max} = s$
 $s_{tp0}^{max} = GetMaxTP_0(b_0, x_s, x_e)$
 $s_{tp1}^{min} = GetMinTP_1(r_e, x_s, x_e)$
 $\mathcal{B}_{0:s} = CacheRepresentation(b_0, x_s, x_e)$
for $s_{tp0} = s_{tp0}^{min} : s_{res} : s_{tp0}^{max}$ **do**
 for $s_{tp1} = s_{tp1}^{min} : s_{res} : s_{tp1}^{max}$ **do**
 $\mathcal{B}_{s_{tp0}:s_{tp1}} = RepUseList(b_0, s_{tp0}, s_{tp1})$
 $\mathcal{V}_{safe} = VSafe(b_0, \mathcal{B}_{s_{tp0}:s_{tp1}})$
 $[valid, \sigma_{min}] = SafeConnect(x_s, x_e, \mathcal{V}_{safe}, s_{tp0}, s_{tp1}, \dot{\psi}, \psi_{path}, a_{max}, v_{def})$
 if $valid \wedge (Time(\sigma_{min}) \leq mintime)$ **then**
 $\sigma_f = \sigma_{min}$
 $mintime = Time(\sigma_f)$
 end
 end
end
return σ_f

Algorithm 16: ComputeSafeDMatrix

Input: $G = [V, E], \psi, a_{max}, v_{def}, b_0, r_e$
Output: Safe D Matrix assuming path independence
for $x_s \in V$ **do**
 for $x_e \in V$ **do**
 $\psi_{path} = atan2((x_e - x_s).y, (x_e - x_s).x)$
 $D[x_s, x_e] = Time(OptSafeConnect(G, x_s, x_e, \psi_{path}, \dot{\psi}, a_{max}, v_{def}, b_0, r_e))$
 end
end
return σ_f

Algorithm 17: OptimizeRoute

Input: $G = [V, E], r, \psi, a_{max}, v_{def}, b_0, r_e$
Output: Route with optimized velocity and heading
 $\sigma = empty$
 $\mathcal{W}_R^{path} = \mathcal{W}_R$
for $i = 0 : (size(r) - 2)$ **do**
 $x_s = r_i$ // r_i is the i^{th} element of the route
 $x_e = r_{i+1}$
 $\sigma_i = OptSafeConnect(G, x_s, x_e, \psi_{path}, \dot{\psi}, a_{max}, v_{def}, b_0^{path}, r_e)$
 $[\sigma_i, b_0^{path}] = OptimizeHeading(\sigma_i, r, b_0^{path}, r_e)$ // PASP (chapter 4)
 $\sigma = \sigma + \sigma_i$
end
return σ

Algorithm 18: SafeRAOr-Greedy(SafeRAOr-G)

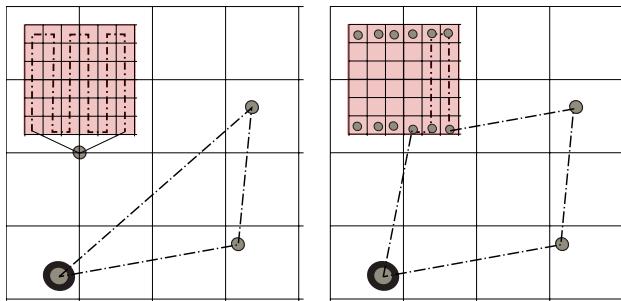
Input: $G = [V, E], v_s, v_e, B, T_r, D_{safe}, b_0$
Output: The best trajectory found in run-time T_r
 $r_g = TSP(v_s, v_e)$ // Seed local search if no global solution is found
 $R = r_g$
if $RouteLength(r_g) > B$ **then**
 return \emptyset
end
 $S = SampleSet(V, v_s, v_e)$
 $r = TSP(S, v_s, v_e)$
if $RouteLength(r) \leq B$ **then**
 $R = R \cup r$
end
for $i = 1 : \exists |V| \wedge runtime < T_r$ **do**
 $v_{new} = WeightedSample(V)$
 if $IsInRoute(r, v_{new})$ **then**
 $r = DeleteFromRoute(r, v_{new})$
 end
 else
 $r = AddToRoute(r, v_{new})$
 end
 if $RouteLength(r) \leq B$ **then**
 $R = R \cup r$
 end
 $R = GreedyLocalSearch(G, v_{new}, R, B)$
end
 $r = \arg \max_{r \in R} I(R)$
 $\sigma = OptimizeRoute(G, r, \psi, a_{max}, v_{def}, b_0, r_e)$
return σ

independence of path cost from motion history effective? Since, for the first part of the *SafeRAOr-G* algorithm, the time of traversing the edges of the graph are assumed to be independent, it needs to be characterized when this assumption starts adversely affecting the performance of the algorithm.

2. *Is there an optimality bound that this algorithm can offer?* On the face of it the algorithm provides a near-optimal solution if the path independence holds true, however it is unclear if the optimality bounds exist if the assumption is invalid.
3. *Is the performance significantly better than that of other sampling based techniques, if yes under what conditions?* Sampling based planners like the ones developed in [Bircher et al., 2016] and [Mascarich et al., 2018] can be used to address the safe, informative path planning problems. It is unclear what kind of problems are better suited to be solved using which algorithm.

Multi-Heuristic Global Path Planning

RAOr has enabled budgeted information gathering to be run online. Given the run-times of the algorithm we are forced to abstract data gathering tasks for an object to a single node, figure 7.5. Although this abstraction helps us reduce the problem size, allowing *RAOr* to run in real-time, the reduced granularity of reasoning about information gathering might lead to sub-optimal behavior.



For example, Figure 7.5 shows a scenario where the vehicle is scouting for landing zones for a helicopter, if the whole landing zone is abstracted as a single node, the node is not visited as the cost of visiting the node exceeds the traveling budget. A faster global planning algorithm would allow for partial coverage of landing zone, hence better performance on data gathering.

We can use sets of heuristics to decrease global planning run times for *RAOr*, allowing us to solve bigger informative path planning algorithms on-board the vehicle.

We presented preliminary results for using heuristics to speed up *RAOr* in chapter 5. However, it is hard to design a single heuristic function that works

Figure 7.5: Scouting of Landing Zones (LZ). We want the vehicle to be able to scout for landing zones. **Left:** Since we cannot deal with big problems online, we reduce the size of those problems by aggregating many nodes into one. In this case vehicle does not scan the LZ, because the cost of visiting the node is out of budget. **Right:** Faster global planning will enable increased granularity (more nodes) and hence enabling partial coverage of LZ.

in all environments. Multi-heuristic A^* [Narayanan et al., 2015] and our previous work [Choudhury et al., 2015] developed an ensemble of heuristics to improve the performance of a planner solving diverse problems. However, these works fail to take into account the hard run-time constraint on an online algorithm.

Given $RAOr$ can operate with multiple heuristics without the compromising its optimality and completeness properties, we can develop an ensemble of heuristics that maximizes $RAOr$'s expected quality of the solution given a distribution of planning problems while treating run-time as a constraint. Section 7.4 formalizes the heuristic set selection problem and provides a linear-time algorithm to select a near-optimal set of heuristics. The set of heuristics can be applied to $RAOr$ algorithm along with the *GreedyLocalSearch* algorithm 10.

Selecting the Set of Heuristics

Let, $\xi \in \Xi$ be an informative path planning problem. Let, $P(\xi)$ be the probability of our global planner encountering the problem during run time. Let the total run-time available for global planning be B_T , let $h_i \in \mathcal{H}$ be the i^{th} . Let, the maximum run-time for a heuristic h_i for all planning problems $\xi \in \Xi : P(\xi) \geq 0$ be t_{h_i} . Let $R_{\mathcal{H}} : \mathcal{HX}\Xi \rightarrow \mathbb{R}$ be the reward achieved by applying $RAOr$ with a heuristic in \mathcal{H} on a planning problem in Ξ . We want to select a set of heuristics $\mathcal{H}_{online} \subseteq \mathcal{H}$ such that the expected reward achieved by the global planner is maximized, equation 7.1.

$$\arg \max_{\mathcal{H}_{online} \subseteq \mathcal{H}} \mathbb{E} \left[\max_{\forall h_i \in \mathcal{H}_{online}} R(h_i, \xi) \right] \quad (7.1)$$

$$\sum_{h_i \in \mathcal{H}_{online}} t_{h_i} \leq B_T$$

The optimization problem in equation 7.1 is combinatorial optimization problem and is NP-Hard. Since the problem is linear combination of maximization, the function being maximized is monotone and sub-modular with a knapsack constraint. We can find a near-optimal solution to equation 7.1 in linear time as presented in algorithm 19.

Although, multi-heuristic $RAOr$ promises to increase the size of the problem we can tackle online, there are still some open research questions that will need to be answered for an effective multi-heuristic $RAOr$.

1. *What should be the base set of heuristics?* Possible candidates are sets of well established TSP heuristics [Reinelt, 1994], with a plethora of greedy reward or distance based heuristics.
2. *Can we improve heuristic sets and their ordering online?* Tallavajhula et. al. [Tallavajhula and Choudhury, 2015] presented methodologies to learn a

Algorithm 19: Greedy Heuristic Set Selection

Input: $\xi, P(\xi), \mathcal{H}, t_h \forall h \in \mathcal{H}, B_T, R_{\mathcal{H}}(\cdot)$
Output: Near-optimal set of heuristics in \mathcal{H}

$\mathcal{H}_{online} = \emptyset$
 $B_{\mathcal{H}} = 0$

while $\mathcal{H} \neq \emptyset$ **do**

$$h^* = \arg \max_{h \in \mathcal{H}} \frac{\mathbb{E}_{P(\xi)} \left[\max_{h \in \mathcal{H}_{online} \cup h} R_{\mathcal{H}}(h_j, \xi) - \max_{h_i \in \mathcal{H}_{online}} R_{\mathcal{H}}(h_i, \xi) \right]}{t_h}$$

if $B_{\mathcal{H}} + t_{h^*} \leq B_T$ **then**

$$\mathcal{H}_{online} = \mathcal{H}_{online} \cup h^*$$

$$B_{\mathcal{H}} = B_{\mathcal{H}} + t_{h^*}$$

end

$\mathcal{H} = \mathcal{H} \setminus h^*$

end

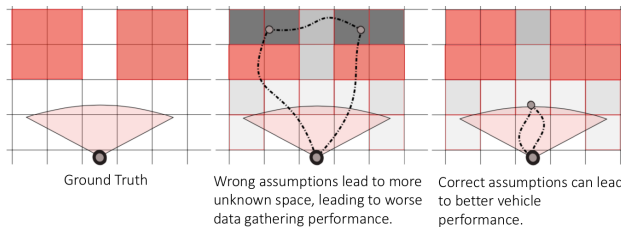
return \mathcal{H}_{online}

set contextual heuristics online, demonstrating significant improvement in planning run-times.

3. *Can we model heuristic interdependence?* Since heuristics affect the set of paths available, they can affect each others performance. It is not clear whether this phenomenon needs to be modeled. If it turns out to be a significant contributor to planning time speed-ups, a learned policy on heuristic sets along the lines of Sanjiban et. al. [Choudhury et al., 2016] might be useful.

The heuristic selection algorithm will allow the set of heuristics to be optimized for the planning problems the vehicle will face during its lifetime. Multi-heuristic *RAO*r with heuristic selected through list selection will allow global planning to run faster enabling the vehicle to solve larger informative path planning problems online.

Data Gathering for Improving Long-Term Deployment



Assumptions are made by data gathering systems for planning and operating effectively under uncertainty. The validity of these assumptions can affect the performance of a data gathering system. Figure 7.6 presents an example where the robot is using Monocular Semantic Mapping for creating the map of the environment. Wrong assumptions about the environment lead to worse mission performance. If the vehicle is deployed for a long term, the cumu-

Figure 7.6: Illustrated here is a scenario where wrong assumptions about the environment lead to an uncertain map, which requires the vehicle to travel more to gain information about the environment. The vehicle gains the same information about the environment with a much smaller path if its assumptions about the object heights in the environment are correct.

lative performance of the vehicle over its deployment might be significantly improved by improving its assumptions about the environment.

Ideally, the information gathering system should be able to gain information for improving its assumptions about the environment. However, there is an exploitation and exploration trade-off involved in deciding for when to gain information for what assumptions. Lets assume we have deployed the information gathering system with Monocular Semantic Mapping. Assuming the system is uncertain about the height of one of the classes of objects it sees in the environment. Obviously this uncertainty affects the systems performance, Figure 7.6. We assume that the robot is deployed at a site for multiple data gathering missions and data collected during the mission is used to update the assumptions that the robot uses for the next mission. The information gathering system should lay emphasis on removing uncertainty from assumptions, if the vehicle has to be deployed for a long time, so that its performance is improved for future missions but if the vehicle is deployed for only one mission there might be very little incentive in gathering data for improving assumptions. Our current formulation does not allow the information gathering agent to reason to do such reasoning.

We define data gathering for improving vehicle's assumptions about the environment as implicit data gathering for assumption improvement.

In this section we develop a formulation to enable implicit data gathering for assumption improvement while exploiting the framework developed for safe efficient data gathering.

Let, the vehicle be deployed for $T \in \mathbb{N}$ number information gathering missions. Let, $a \in \mathcal{A}$ be the vector of assumptions. Let, $b^a \in \mathcal{B}_{\mathcal{A}}$ be the probability distribution over the assumptions, $b^a : \mathcal{A} \rightarrow [0, 1]$. The observation model of the robot depends on the assumptions it makes about the environment, let the observation model be given by $P_{b^a}(o|m)$, where $o \in \mathcal{O}$ is an observation and $m \in \mathcal{M}$ is the state of the environment. Let $o_n^a \in \mathcal{O}^a$ be the observations taken by sensors during the n^{th} deployment. Let $P_{\mathcal{A}}(o^a|m)$ be observation model used to update assumptions. For the n^{th} mission the vehicle selects a route $r_n^* \subset V$ to optimize the information theoretic reward function $I(\cdot) + I_{w_n}(\cdot) : V \rightarrow \mathbb{R}$. Where, I_{w_n} is a reward function that encourages gathering of data for improving assumptions depending on w_n . $w_n \in \mathbb{R}^{|\mathcal{A}|}$ is a real vector of same dimension as space \mathcal{A} and decides how important each assumption is for data gathering. Hence we want to find a policy $\sigma_a : \mathcal{B}_{\mathcal{A}} \rightarrow \mathbb{R}^{|\mathcal{A}|}$, such that the explicit information demanded by the user is maximized over T runs.

$$\sigma_a^* = \arg \max_{\sigma_a} \sum_{n=0:T} \mathbb{E} I(r_n^*) \quad (7.2)$$

$$\begin{aligned}
b_n^a(a) &= \alpha P_{\mathcal{A}}(o^a|m)b_{n-1}^a(m) \\
r_n^* &= \arg \max_r I(r) + I(w_n)(r) \\
C(r_n^*) &\leq B
\end{aligned}$$

Where, α denotes a constant normalizer.

Equation 7.2 presents a formulation to address the data gathering problem for long-term improvement. However, there are plenty of open open questions that need to be addressed before this formulation is put in practice.

1. *How do we restrict the space of assumptions?* Space of assumptions can theoretically be as big as assumptions about all the objects in the world. Given the uncertainty in the location and kind of deployment of the system can we reduce that space?
2. *What is the correct distribution to represent belief in assumption space?*
3. *How do solve the daunting optimization problem?* We can use Monte-Carlo Tree Searches (MCTS) for solving the optimization problem. MCTS is a popular technique for solving large POMDPs [Marchant et al., 2014]. It turns a tedious search in decision trees into an efficient approximation using Monte-Carlo samples from the tree. Its main advantage over other techniques, such as Point-Based Value Iteration is that it does not require the overhead of maintaining alpha-functions over all states nor choosing the states for which alpha-functions should be maintained.

7.5 Final Thoughts

The long-term vision of this work is to enable autonomous agents to actively reason about their sensory and dynamic capabilities while being safe. The agents should be able to efficiently use their sensory resources to gather implicit and explicit information pertinent to their mission. The algorithms presented in the thesis, form a framework for safe, efficient, multi-resolution data gathering that has enabled UAVs to operate in diverse environments, scales, and applications. We evaluated our algorithms on multiple UAVs varying from full-scale helicopters to small quad-rotors, running closed-loop autonomous missions that cumulatively span hundreds of kilometers. We have demonstrated that contributions, and insights regarding safety, implicit and explicit data gathering enable agents to partially achieve our vision. The future work section lists concrete steps in furthering the development of such autonomous agents. We hope that insights developed in this work help autonomous agents to reach our long-term vision.

Bibliography

Nerian SPI Stereo Vision System. URL nerian.com.

Pieter Abbeel, Adam Coates, Morgan Quigley, and Andrew Y Ng. An application of reinforcement learning to aerobatic helicopter flight. In *Advances in neural information processing systems*, pages 1–8, 2007.

Karim Abdel-Malek, Yingzhou Yang, Denis Blackmore, and Ken Joy. Swept volumes: Foundations, perspectives, and applications. *Int. J. Shape Modeling*, 12(1):87–127, 2006.

F Adolf and J. S. Dittrich. Evaluation of the ARTIS sampling-based path planner using an obstacle field navigation benchmark. In *Proceedings of the American Helicopter Society 68th Annual Forum. Ft. Worth, TX*, 2012.

S. Arora and S. Scherer. Pasp: Policy based approach for sensor planning. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3479–3486, May 2015. DOI: 10.1109/ICRA.2015.7139680.

Sankalp Arora and Sebastian Scherer. Randomized algorithm for informative path planning. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*. IEEE, 2017.

Sankalp Arora, Sezal Jain, Sebastian Scherer, Stephen Nuske, Lyle Chamberlain, and Sanjiv Singh. Infrastructure-free shipdeck tracking for autonomous landing. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 323–330. IEEE, 2013.

Sankalp Arora, Sanjiban Choudhury, Sebastian Scherer, and Daniel Althoff. A principled approach to enable safe and high performance maneuvers for autonomous rotorcraft. In *American Helicopter Society 70th Annual Forum*. AHS, 2014.

Sankalp Arora, Sanjiban Choudhury, Daniel Althoff, and Sebastian Scherer. Emergency maneuver library-ensuring safe navigation in partially known environments. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pages 6431–6438. IEEE, 2015.

Sankalp Arora, Sanjiban Choudhury, and Sebastian Scherer. Hindsight is only 50/50: Unsuitability of mdp based approximate pomdp solvers for multi-resolution information gathering. *arXiv preprint arXiv:1804.02573*, 2018.

Ruzena Bajcsy. Active perception. 1988.

Bram Bakker, Zoran Zivkovic, and Ben Krose. Hierarchical dynamic programming for robot path planning. In *Intelligent Robots and Systems, 2005.(IROS 2005). 2005 IEEE/RSJ International Conference on*, pages 2756–2761. IEEE, 2005.

Dana H Ballard. Animate vision uses object-centered reference frames. In *Advanced neural computers*, pages 229–236. Elsevier, 1990.

Frederick J Beutler. Dynamic programming: Deterministic and stochastic models (dimitri p. bertsekas). *SIAM Review*, 31(1):132, 1989.

Cooper Bills, Joyce Chen, and Ashutosh Saxena. Autonomous mav flight in indoor environments using single image perspective cues. In *Robotics and automation (ICRA), 2011 IEEE international conference on*, pages 5776–5783. IEEE, 2011.

Jonathan Binney, Andreas Krause, and Gaurav S Sukhatme. Optimizing waypoints for monitoring spatiotemporal phenomena. *The International Journal of Robotics Research*, 32(8):873–888, 2013.

Andreas Bircher, Kostas Alexis, Michael Burri, Philipp Oettershagen, Sammy Omari, Thomas Mantel, and Roland Siegwart. Structural inspection path planning via iterative viewpoint resampling with application to aerial robotics. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pages 6423–6430. IEEE, 2015.

Andreas Bircher, Mina Kamel, Kostas Alexis, Helen Oleynikova, and Roland Siegwart. "Receding horizon" next-best-view" planner for 3d exploration. In *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, pages 1462–1468. IEEE, 2016.

F. Blachini. Set invariance in control. *Automatica*, 35(11):1747–1767, 1999.

Frederic Bourgault, Alexei A Makarenko, Stefan B Williams, Ben Grocholsky, and Hugh F Durrant-Whyte. Information based adaptive robotic exploration. In *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on*, volume 1, pages 540–545. IEEE, 2002.

Frederic Bourgault, Tomonari Furukawa, and Hugh F Durrant-Whyte. Coordinated decentralized search for a lost target in a bayesian world. In *Intelligent Robots and Systems, 2003.(IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, volume 1, pages 48–53. IEEE, 2003.

- Michael S Branicky, Ross A Knepper, and James J Kuffner. Path and trajectory diversity: Theory and algorithms. In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pages 1359–1364. IEEE, 2008.
- Markus Buchberger, K-W Jorg, and Ewald von Puttkamer. Laserradar and sonar based world modeling and motion control for fast obstacle avoidance of the autonomous mobile robot MOBOT-IV. In *Robotics and Automation, 1993. Proceedings., 1993 IEEE International Conference on*, pages 534–540. IEEE, 1993.
- Alec Cameron and Hugh Durrant-Whyte. A bayesian approach to optimal sensor placement. *The International Journal of Robotics Research*, 9(5): 70–88, 1990.
- Benjamin Charrow, Sikang Liu, Vijay Kumar, and Nathan Michael. Information-theoretic mapping using cauchy-schwarz quadratic mutual information. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pages 4791–4798. IEEE, 2015.
- Chandra Chekuri and Martin Pal. A recursive greedy algorithm for walks in directed graphs. In *Foundations of Computer Science*. IEEE, 2005.
- Chandra Chekuri, Nitish Korula, and Martin Pál. Improved algorithms for orienteering and related problems. *ACM Transactions on Algorithms (TALG)*, 8(3):23, 2012.
- Chen CHEN, Shih-Fen Cheng, and Hoong Chuin Lau. Multi-agent orienteering problem with time-dependent capacity constraints. *Web Intelligence and Agent Systems*, 2014.
- Min Chen, Emilio Frazzoli, David Hsu, and Wee Sun Lee. Pomdp-lite for robust robot planning under uncertainty. In *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, pages 5427–5433. IEEE, 2016.
- Shengyong Chen, Youfu Li, and Ngai Ming Kwok. Active vision in robotic systems: A survey of recent developments. *The International Journal of Robotics Research*, 30(11):1343–1377, 2011.
- S. Choudhury, S. Arora, and S. Scherer. The planner ensemble and trajectory executive: A high performance motion planning system with guaranteed safety. In *AHS Forum 70*, 2014.
- S. Choudhury, A. Kapoor, G. Ranade, and D. Dey. Learning to gather information via imitation. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 908–915, May 2017a. DOI: 10.1109/ICRA.2017.7989112.

Sanjiban Choudhury, Sankalp Arora, and Sebastian Scherer. The planner ensemble: Motion planning by executing diverse algorithms. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pages 2389–2395. IEEE, 2015.

Sanjiban Choudhury, Ashish Kapoor, Gireeja Ranade, and Debadeepta Dey. Learning to gather information via imitation. *arXiv preprint arXiv:1611.04180*, 2016.

Sanjiban Choudhury, Mohak Bhardwaj, Sankalp Arora, Ashish Kapoor, Gireeja Ranade, Sebastian Scherer, and Debadeepta Dey. Data-driven planning via imitation learning. *arXiv preprint arXiv:1711.06391*, 2017b.

Nicos Christofides. Worst-case analysis of a new heuristic for the travelling salesman problem. Technical report, DTIC Document, 1976a.

Nicos Christofides. Worst-case analysis of a new heuristic for the travelling salesman problem. Technical report, DTIC Document, 1976b.

Joachim Denzler and Christopher M Brown. Information theoretic sensor data selection for active object recognition and state estimation. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (2):145–157, 2002.

Debadeepta Dey, Tian Yu Liu, Boris Sofman, and Drew Bagnell. Efficient optimization of control libraries. Technical report, 2011.

Geetesh Dubey, Sankalp Arora, and Sebastian Scherer. Droan—disparity-space representation for obstacle avoidance. In *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*, pages 1324–1330. IEEE, 2017.

J. J. Enright, E. Frazzoli, K. Savla, and F. Bullo. On multiple UAV routing with stochastic targets: Performance bounds and algorithms. In *Proceedings AIAA Guidance, Navigation, and Control Conference and Exhibit, Aug. 2005*.

Lawrence H Erickson and Steven M LaValle. Survivability: Measuring and ensuring path diversity. In *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pages 2068–2073. IEEE, 2009.

William Feller. An introduction to probability theory and its applications. vol. i. 1950.

Dieter Fox, Wolfram Burgard, and Sebastian Thrun. The dynamic window approach to collision avoidance. *IEEE Robotics & Automation Magazine*, 4 (1):23–33, 1997.

T. Fraichard and H. Asama. Inevitable collision states. A step towards safer robots? *Advanced Robotics*, 18(10):1001–1024, 2004a.

- Thierry Fraichard and Hajime Asama. Inevitable collision states—a step towards safer robots? *Advanced Robotics*, 18(10):1001–1024, 2004b.
- E. Frazzoli, M. A. Dahleh, and E. Feron. Real-time motion planning for agile autonomous vehicles. *AIAA Journal of Guidance and Control*, 25(1): 116–129, 2002.
- Paul Furgale and Timothy D Barfoot. Visual teach and repeat for long-range rover autonomy. *Journal of Field Robotics*, 27(5):534–560, 2010.
- Jonathan D. Gammell, Siddhartha S. Srinivasa, and Timothy D. Barfoot. Bit*: Batch informed trees for optimal sampling-based planning via dynamic programming on implicit random geometric graphs. *CoRR*, abs/1405.5848, 2014. URL <http://arxiv.org/abs/1405.5848>.
- James J Gibson. *The perception of the visual world*. 1950.
- C Goerzen and M Whalley. Minimal risk motion planning: a new planner for autonomous UAVs in uncertain environment. In *AHS International Specialists Meeting on Unmanned Rotorcraft, Tempe, Arizona*, 2011.
- Pascal Gohl, Dominik Honegger, Sammy Omari, Markus Achtelik, Marc Pollefeys, and Roland Siegwart. Omnidirectional visual obstacle detection using embedded FPGA. *IEEE International Conference on Intelligent Robots and Systems*, 2015-Decem:3938–3943, 2015. ISSN 21530866. DOI: 10.1109/IROS.2015.7353931.
- Bruce L Golden, Larry Levy, and Rakesh Vohra. The orienteering problem. *Naval Research Logistics (NRL)*, 34(3):307–318, 1987.
- Michael A. Goodrich, Bryan S. Morse, Damon Gerhardt, Joseph L. Cooper, Morgan Quigley, Julie A. Adams, and Curtis Humphrey. Supporting wilderness search and rescue using a camera-equipped mini uav. *Journal of Field Robotics*, 25(1-2):89–110, 2008. ISSN 1556-4967. DOI: 10.1002/rob.20226. URL <http://dx.doi.org/10.1002/rob.20226>.
- Green, Colin J. and Alonzo Kelly. Toward optimal sampling in the space of paths. pages 281–292, 2011.
- Ruijie He, Emma Brunskill, and Nicholas Roy. Puma: Planning under uncertainty with macro-actions. In *AAAI*, 2010.
- Jarrold C Hodgson, Shane M Baylis, Rowan Mott, Ashley Herrod, and Rohan H Clarke. Precision wildlife monitoring using unmanned aerial vehicles. *Scientific reports*, 6, 2016.
- Gabriel M Hoffmann, Steven L Waslander, and Claire J Tomlin. Quadrotor helicopter trajectory tracking control. In *AIAA guidance, navigation and control conference and exhibit*, pages 1–14, 2008.

Geoffrey Hollinger, Sanjiv Singh, Joseph Djughash, and Athanasios Kehagias. Efficient multi-robot search for a moving target. *The International Journal of Robotics Research*, 28(2):201–219, 2009.

Geoffrey A Hollinger. Long-horizon robotic search and classification using sampling-based motion planning.

Kaijen Hsiao, Tomás Lozano-Pérez, and Leslie Pack Kaelbling. Robust belief-based execution of manipulation programs. In *Eighth Intl. Workshop on the Algorithmic Foundations of Robotics*. Citeseer, 2008.

Albert S Huang, Abraham Bachrach, Peter Henry, Michael Krainin, Daniel Maturana, Dieter Fox, and Nicholas Roy. Visual odometry and mapping for autonomous flight using an rgb-d camera. In *International Symposium on Robotics Research (ISRR)*, pages 1–16, 2011.

Sezal Jain, Stephen Nuske, Andrew Chambers, Luke Yoder, Hugh Cover, Lyle Chamberlain, Sebastian Scherer, and Sanjiv Singh. Autonomous river exploration. In *Field and Service Robotics*, pages 93–106. Springer, 2015.

Shervin Javdani, Siddhartha S Srinivasa, and J Andrew Bagnell. Shared autonomy via hindsight optimization. *arXiv preprint arXiv:1503.07619*, 2015.

Heather L Jones, Uland Wong, Kevin M Peterson, Jason Koenig, Aashish Sheshadri, and William L Red Whittaker. Complementary flyover and rover sensing for superior modeling of planetary features. In *Field and Service Robotics*, pages 415–429. Springer, 2014.

Brian J. Julian, Sertac Karaman, and Daniela Rus. On mutual information-based control of range sensing robots for mapping applications. *The International Journal of Robotics Research*, 33(10):1375–1392, 2014. DOI: 10.1177/0278364914526288.

Leslie Pack Kaelbling, Michael L Littman, and Anthony R Cassandra. Planning and acting in partially observable stochastic domains. *Artificial intelligence*, 101(1-2):99–134, 1998.

Michael C Koval, David Hsu, Nancy S Pollard, and Siddhartha S Srinivasa. Configuration lattices for planar contact manipulation under uncertainty. *arXiv preprint arXiv:1605.00169*, 2016a.

Michael C Koval, Nancy S Pollard, and Siddhartha S Srinivasa. Pre-and post-contact policy decomposition for planar contact manipulation under uncertainty. *The International Journal of Robotics Research*, 35(1-3): 244–264, 2016b.

Andreas Krause. *Optimizing Sensing: Theory and Applications*. PhD thesis, Carnegie Mellon University, December 2008.

Andreas Krause and Carlos Guestrin. Near-optimal observation selection using submodular functions. In *AAAI*, volume 7, pages 1650–1654, 2007.

Steven M LaValle. *Planning algorithms*. Cambridge university press, 2006.

Zhan Wei Lim, David Hsu, and Wee Sun Lee. Adaptive informative path planning in metric spaces. *The International Journal of Robotics Research*, 35(5):585–598, 2016.

Iain Little, Sylvie Thiebaux, et al. Probabilistic planning vs. replanning. In *ICAPS Workshop on IPC: Past, Present and Future*, 2007.

Michael L Littman, Anthony R Cassandra, and Leslie Pack Kaelbling. Learning policies for partially observable environments: Scaling up. In *Machine Learning Proceedings 1995*, pages 362–370. Elsevier, 1995a.

Michael L. Littman, Thomas L. Dean, and Leslie Pack Kaelbling. On the complexity of solving markov decision problems. In *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence, UAI'95*, pages 394–402, San Francisco, CA, USA, 1995b. Morgan Kaufmann Publishers Inc. ISBN 1-55860-385-9. URL <http://dl.acm.org/citation.cfm?id=2074158.2074203>.

Roman Marchant, Fabio Ramos, Scott Sanner, et al. Sequential bayesian optimisation for spatial-temporal monitoring. In *UAI*, pages 553–562, 2014.

Frank Mascarich, Taylor Wilson, Christos Papachristos, and Kostas Alexis. Radiation source localization in gps-denied environments using aerial robots. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6537–6544. IEEE, 2018.

L. Matthies, R. Brockers, Y. Kuwata, and S. Weiss. Stereo vision-based obstacle avoidance for micro air vehicles using disparity space. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3242–3249, May 2014. DOI: 10.1109/ICRA.2014.6907325.

Daniel Maturana, Sankalp Arora, and Sebastian Scherer. Looking forward: A semantic mapping system for scouting with micro-aerial vehicles. In *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*, pages 6691–6698. IEEE, 2017.

B Mettler, Z Kong, C Goerzen, and M Whalley. Benchmarking of obstacle field navigation algorithms for autonomous helicopters. In *Proceedings of the American Helicopter Society 66th Annual Forum*. Phoenix, AZ, 2010.

H. Michalska and D. Q. Mayne. Robust receding horizon control of constrained systems. *IEEE Transactions on Automatic Control*, 38(11): 1623–1633, 1993.

Rajeev Motwani and Prabhakar Raghavan. *Randomized algorithms*. Chapman & Hall/CRC, 2010.

V Myers and DP Williams. A pomdp for multi-view target classification with an autonomous underwater vehicle. In *OCEANS 2010*, pages 1–5. IEEE, 2010.

Venkatraman Narayanan, Sandip Aine, and Maxim Likhachev. Improved multi-heuristic a* for searching with uncalibrated heuristics. In *Eighth Annual Symposium on Combinatorial Search*, 2015.

G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher. An analysis of approximations for maximizing submodular set functions—I. *Mathematical Programming*, 14(1):265–294, 1978.

Christos Papadimitriou and John N. Tsitsiklis. The complexity of markov decision processes. *Math. Oper. Res.*, 12(3):441–450, August 1987. ISSN 0364-765X. DOI: 10.1287/moor.12.3.441. URL <http://dx.doi.org/10.1287/moor.12.3.441>.

Joelle Pineau, Geoff Gordon, Sebastian Thrun, et al. Point-based value iteration: An anytime algorithm for pomdps. In *IJCAI*, volume 3, pages 1025–1032, 2003.

Mihail Pivtoraiko and Alonzo Kelly. Kinodynamic motion planning with state lattice motion primitives. In *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pages 2172–2179. IEEE, 2011.

Patrick A Plonski and Volkan Isler. Approximation algorithms for tours of height-varying view cones. *The International Journal of Robotics Research*, page 0278364918784353, 2016.

Gerhard Reinelt. *The traveling salesman: computational solutions for TSP applications*. Springer-Verlag, 1994.

Stéphane Ross, Joelle Pineau, Sébastien Paquet, and Brahim Chaib-Draa. Online planning algorithms for pomdps. *Journal of Artificial Intelligence Research*, 32:663–704, 2008.

Allison Ryan and J Karl Hedrick. Particle filter based information-theoretic active sensing. *Robotics and Autonomous Systems*, 58(5):574–584, 2010.

Sebastian Scherer, Lyle Chamberlain, and Sanjiv Singh. Autonomous landing at unprepared sites by a full-scale helicopter. *Robotics and Autonomous Systems*, 60(12):1545–1562, 2012a.

Sebastian Scherer, Joern Rehder, Supreeth Achar, Hugh Cover, Andrew Chambers, Stephen Nuske, and Sanjiv Singh. River mapping from a flying

robot: state estimation, river detection, and obstacle mapping. *Autonomous Robots*, 33(1-2):189–214, 2012b.

Sebastian Scherer, Joern Rehder, Supreeth Achar, Hugh Cover, Andrew Chambers, Stephen Nuske, and Sanjiv Singh. River mapping from a flying robot: state estimation, river detection, and obstacle mapping. *Autonomous Robots*, 32(5):1–26, 2012c.

Uwe Schöning. A probabilistic algorithm for k-sat and constraint satisfaction problems. In *Foundations of Computer Science, 1999. 40th Annual Symposium on*, pages 410–414. IEEE, 1999a.

Uwe Schöning. A probabilistic algorithm for k-sat and constraint satisfaction problems. In *Foundations of Computer Science, 1999. 40th Annual Symposium on*, pages 410–414. IEEE, 1999b.

Tom Schouwenaars, Jonathan How, and Eric Feron. Receding horizon path planning with implicit safety guarantees. In *American Control Conference, 2004. Proceedings of the 2004*, volume 6, pages 5576–5581. IEEE, 2004.

Guy Shani, Joelle Pineau, and Robert Kaplow. A survey of point-based pomdp solvers. *Autonomous Agents and Multi-Agent Systems*, 27(1):1–51, 2013.

Claude E Shannon. Prediction and entropy of printed english. *Bell system technical journal*, 30(1):50–64, 1951.

Amarjeet Singh, Andreas Krause, Carlos Guestrin, and William J Kaiser. Efficient informative sensing using multiple robots. *Journal of Artificial Intelligence Research*, pages 707–755, 2009.

J. W. Langelaan S.Q. Marlow . Local terrain mapping for obstacle avoidance using monocular vision. In *AHS Unmanned Vehicles Specialist's Forum, AHS*, 2009.

Cyrill Stachniss, Giorgio Grisetti, and Wolfram Burgard. Information gain-based exploration using rao-blackwellized particle filters. In *Robotics: Science and Systems*, volume 2, pages 65–72, 2005.

Matthew Streeter and Daniel Golovin. An online algorithm for maximizing submodular functions. Technical report, 2007.

Abhijeet Tallavajhula and Sanjiban Choudhury. List prediction for motion planning: Case studies. *Robotics Institute, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-15-25*, 2015.

A. Tamar, G. Thomas, T. Zhang, S. Levine, and P. Abbeel. Learning from the hindsight plan – episodic mpc improvement. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 336–343, May 2017. DOI: 10.1109/ICRA.2017.7989043.

- Jay Martin Tenenbaum. Accommodation in computer vision. Technical report, Stanford Univ Ca Dept of Computer Science, 1970.
- Sebastian Thrun. Learning occupancy grid maps with forward sensor models. *Autonomous robots*, 15(2):111–127, 2003.
- Peter Trautman and Andreas Krause. Unfreezing the robot: Navigation in dense, interacting crowds. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 797–803. IEEE, 2010.
- Jur Van Den Berg, Sachin Patil, and Ron Alterovitz. Motion planning under uncertainty using iterative local optimization in belief space. *The International Journal of Robotics Research*, 31(11):1263–1278, 2012.
- Pieter Vansteenwegen, Wouter Souffriau, and Dirk Van Oudheusden. The orienteering problem: A survey. *European Journal of Operational Research*, 209(1):1–10, 2011.
- Abraham Wald. Sequential tests of statistical hypotheses. *The Annals of Mathematical Statistics*, 16(2):117–186, 1945.
- David Wettergreen, Greydon Foil, Michael Furlong, and David R Thompson. Science autonomy for rover subsurface exploration of the atacama desert. *AI Magazine*, 35(4):47–60, 2014.
- Brian Yamauchi. A frontier-based approach for autonomous exploration. In *Computational Intelligence in Robotics and Automation, 1997. CIRA'97., Proceedings., 1997 IEEE International Symposium on*, pages 146–151. IEEE, 1997.
- Luke Yoder and Sebastian Scherer. Autonomous exploration for infrastructure modeling with a micro aerial vehicle. In *Field and service robotics*, pages 427–440. Springer, 2016.
- Sung Wook Yoon, Alan Fern, Robert Givan, and Subbarao Kambhampati. Probabilistic planning via determinization in hindsight. In *AAAI*, pages 1010–1016, 2008.
- Jingjin Yu, Mac Schwager, and Daniela Rus. Correlated orienteering problem and its application to informative path planning for persistent monitoring tasks. In *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pages 342–349. IEEE, 2014.
- Haifeng Zhang and Yevgeniy Vorobeychik. Submodular optimization with routing constraints. In *AAAI*, pages 819–826, 2016.
- Tianhao Zhang, Gregory Kahn, Sergey Levine, and Pieter Abbeel. Learning deep control policies for autonomous aerial vehicles with mpc-guided policy search. In *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, pages 528–535. IEEE, 2016.