

Real-Time Motion Planning and Safe Navigation in Dynamic Multi-Robot Environments

James Robert Bruce

CMU-CS-06-181

December 15, 2006

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee:

Manuela Veloso, Chair

Avrim Blum

James Kuffner

Tony Stentz

Lydia Kavraki (Rice University)

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

Copyright © 2006 James Robert Bruce

This research was sponsored in part by United States Grants Nos. DABT63-99-1-0013, F30602-98-2-0135, F30602-97-2-0250, and NBCH-1040007. It was also supported by Rockwell Scientific Co., LLC under subcontract No. B4U528968 and prime contract No. W911W6-04-C-0058 with the US Army. The views and conclusions contained herein are those of the author, and do not necessarily reflect the position or policy of the sponsoring institutions, and no official endorsement should be inferred.

Keywords: Robotics, Robot Navigation, Motion Planning, Path Planning, Multi-Robot Systems, Robotic Soccer

Abstract

All mobile robots share the need to navigate, creating the problem of motion planning. In multi-robot domains with agents acting in parallel, highly complex and unpredictable dynamics can arise. This leads to the need for navigation calculations to be carried out within tight time constraints, so that they can be applied before the dynamics of the environment make the calculated answer obsolete. At the same time, we want the robots to navigate robustly and operate safely without collisions. While motion planning has been used for high level robot navigation, or limited to semi-static or single-robot domains, it has often been dismissed for the real-time low-level control of agents due to the limited computational time and the unpredictable dynamics. Many robots now rely on local reactive methods for immediate control of the robot, but if the reason for avoiding motion planning is execution speed, the answer is to find planners that can meet this requirement. Recent advances in traditional path planning algorithms may offer hope in resolving this type of scalability, if they can be adapted to deal with the specific problems and constraints mobile robots face. Also, in order to maintain safety, new scalable methods for maintaining collision avoidance among multiple robots are needed in order to free motion planners from the “curse of dimensionality” when considering the safety of multiple robots with realistic physical dynamics constraints. This thesis contributes the pairing of real-time motion planning which builds on existing modern path planners, and a novel cooperative dynamics safety algorithm for high speed navigation of multiple agents in dynamic domains. It also explores near real-time kinematically limited motion planning for more complex environments. The thesis algorithms have been fully implemented and tested with success on multiple real robot platforms.

Acknowledgements

First, I would like to thank my parents for supporting and guiding me as a student and as a son. I would like to thank my advisor, Manuela Veloso for her support of my research since 1998, picking up where my undergraduate education left off. Through much of my time at Carnegie Mellon, I would like to thank Scott Lenser, for always lending an ear for a problem research related or otherwise. The lab hasn't been the same since you left. I am indebted as well to all the members of our RoboCup teams past and present, for coming together to define a collective domain for pushing research which is much greater than the sum of its parts. Particular members who have helped me since I began my work include Mike Bowling, Brett Browning, Tucker Balch, Jennifer Lin, Dinesh Govindaraju, Mike Licitra, and Stefan Zickler. Without your work the teams never would have made it. I would like to thank other fellow lab members for our other collaborations, as well as the willingness to listen to me while I stood in front of a whiteboard to expound on a problem. Douglas Vail, Sonia Chernova, and Colin McMillen have best helped fill that role.

I am also indebted to particular research groups for supporting various aspects of my work. I would like to thank Masahiro Fujita for supporting an internship at Sony Corporation which allowed to explore planning on the QRIO platform. I would like to thank Tadashi Naruse, all his students (in particular Shinya Hibino) of Aichi Prefectural University for keeping our RoboCup project alive with their offer of joint collaboration and hosting me for RoboCup 2005. Most recently, I would like to thank the members of the ACO project from Rockwell Scientific for supporting my research work on complex kinematic planners.

Table of Contents

1	Introduction	23
1.1	Navigation	23
1.2	Dynamics	24
1.3	Problem Definitions	26
1.3.1	Motion Planning	26
1.3.2	Cooperative Safety	28
1.3.3	Complicating Factors	29
1.4	Existing Approaches to Motion Planning	29
1.5	Approach and Thesis	33
1.6	Thesis Contributions	36
1.7	Guide to the Thesis	38
2	Domains	39
2.1	RoboCup Small-Size Multi-Robot Soccer	40

2.2	Fixed Wing UAV	42
2.3	QRIO Humanoid Robot	44
2.4	Summary of Domains	46
3	Collision Detection	49
3.0.1	Contributions of this Chapter	50
3.1	Existing Approaches to Broad-Phase Collision Detection	51
3.1.1	Coordinate Sorting and Sweep-and-Prune	51
3.1.2	Spatial Hashing	52
3.1.3	Spatial Partitioning Trees	53
3.1.4	Hierarchical Bounding Volumes	55
3.2	Approach to Collision Checking	55
3.2.1	Extent Masks	55
3.2.2	Heuristically Balanced AABB tree	59
3.2.3	Narrow-Phase Collision Checking	62
3.2.4	Performance Measurement	65
3.3	Summary	70
4	Motion Planning	71
4.1	Approach	71
4.1.1	Contributions of this Chapter	72

4.2	Existing Planning Methods	73
4.2.1	RRT Algorithm in Detail	73
4.2.2	PRM Algorithm in Detail	74
4.3	Execution Extended RRT (ERRT)	76
4.3.1	Testing ERRT for RoboCup	79
4.3.2	Conclusions in Applying ERRT to Soccer Robot Navigation	82
4.3.3	Exploring waypoint Cache Replacement Strategies	84
4.3.4	Bidirectional Multi-Bridge ERRT	85
4.3.5	Comparing ERRT with the Visibility Graph in 2D	89
4.4	Dynamic PRM	93
4.5	The Abstract Domain Interface	97
4.6	Practical Issues in Planning	99
4.6.1	Simple General Path Smoothing	99
4.6.2	Robust Planning	99
4.7	Applications	101
4.7.1	Fixed Wing UAV	101
4.7.2	QRIO Humanoid Robot Planner	105
5	Safe Navigation	109
5.0.3	Contributions of this Chapter	109

5.1	Robot Model	110
5.2	The Dynamic Window Approach	114
5.3	Dynamics Safety Search	116
5.4	Guarantee of Safety	124
5.5	Improving Efficiency	126
5.6	Evaluation and Results	128
5.6.1	Simulation Evaluation	128
5.6.2	Real Robot Evaluation	134
5.7	Conclusion	137
6	Related Work	139
6.1	Motion Planning	139
6.1.1	Scope and Categorization	140
6.1.2	Graph and Grid Methods	141
6.1.3	Visibility Graph	143
6.1.4	Randomized Path Planner (RPP)	143
6.1.5	Rapidly Exploring Random Trees (RRT)	144
6.1.6	RRT Variants	145
6.1.7	Probabalistic Roadmaps (PRM)	147
6.1.8	PRM Variants	148

6.1.9	Randomized Forward Planners	150
6.2	Safety Methods	151
6.2.1	Dynamic Window	151
6.3	Algorithm Summary	151
A	Machine Vision for Navigation	153
A.1	CMVision: The Color Machine Vision Library	154
A.1.1	Color Image Segmentation	154
A.1.2	Color Spaces	155
A.1.3	Thresholding	158
A.1.4	Connected Regions	160
A.1.5	Extracting Region Information	162
A.1.6	Density-Based Region Merging	162
A.1.7	Performance	163
A.1.8	Summary of the CMVision Library	164
A.2	Pattern Detection	164
A.2.1	Single Patches	167
A.2.2	Patterns and Detection	170
A.2.3	Pattern Comparison	173
A.2.4	Summary of Pattern Vision	176

B The CMDragons Multi-Robot System	179
B.1 Software Overview	182
B.2 Robot Hardware	183
B.3 Robot Model	185
B.4 Motion Control	186
B.5 Objective Assignment for Multi-Robot Planning	188
B.6 One-touch Ball Control	193
B.7 Summary and Results	195

List of Figures

1.1	A qualitative comparison of the level of domain dynamics targeted by the design of various motion planning algorithms.	33
1.2	A qualitative explanation of planning challenges, plotting problem difficulty against the time used for a hypothetical planning algorithm.	34
1.3	A hypothetical comparison of two algorithms under a given timing constraint (A), and a difficulty constraint (B).	34
2.1	Several platforms used in navigational research. These include RoboCup small-size league soccer (left), a fixed wing UAV (center), and the QRIO humanoid robot (right).	39
2.2	The dimensions of the current RoboCup small-size field.	40
2.3	Two teams are shown playing soccer in the RoboCup small size league. . . .	41
2.4	Five generations of Carnegie Mellon robots: (from left) 1997, 1998-99, 2001, 2002-03, 2006	42
2.5	A robot on the left finds a path to a goal on the right using the ERRT algorithm.	43
2.6	A kinodynamically-limited search tree (left), and the corresponding simplified plan (right) for the UAV. The plan length is approximately 13km long. . . .	45

2.7	The Sony QRIO robot (left), and a stereo generated occupancy grid environment map (right).	46
2.8	QRIO on top of an autonomously reached step obstacle.	47
3.1	Examples of KD and BSP spatial partitioning trees	54
3.2	Examples of several space partitioning hierarchies	57
3.3	Example environment with the corresponding projected extent masks	58
3.4	Two broad-phase bounding box checks using projected extent masks	58
3.5	Example swept circle obstacle check using only distance queries.	63
3.6	The first levels of bounding boxes on a terrain K-D tree.	65
3.7	Nodes expanded on an example terrain distance query.	66
3.8	Two domains with 64 obstacles used as collision checking benchmarks.	67
3.9	Collision query time distribution with 64 circular obstacles.	68
3.10	Scaling of collision queries with obstacles for extent masks	68
3.11	Scaling of collision queries with obstacles for AABB tree	69
3.12	Scaling of collision queries with obstacles for a linear array	69
4.1	An example from the simulation-based RRT planner, shown at two times during a run. The tree grows from the initial state. The best plan is shown in bold, and cached waypoints are shown as small black dots.	79

4.2	The effect of waypoints. The lines shows the sorted planning times for 2670 plans with and without waypoints, sorted by time to show the cumulative performance distribution. The red line shows performance without waypoints, while the blue line shows performance with waypoints (Waypoints=50, p[Waypoint]=0.4).	80
4.3	Planning times vs. number of nodes expanded with and without a KD-tree for nearest neighbor lookup. The KD-tree improves performance, with the gap increasing with the number of nodes due to its expected complexity advantage ($E[O(\log(n))]$ vs. $E[O(n)]$ for the naive version).	83
4.4	Comparison of several waypoint cache replacements strategies while varying the waypoint cache selection probability.	85
4.5	The effect of repeated extensions in ERRT on plan length.	87
4.6	The effect of multiple connection points in ERRT on plan length.	88
4.7	Environments used for benchmarking planners.	90
4.8	A comparison of ERRT with the visibility graph method across several 2D domains.	92
4.9	Dynamic PRM connection technique, using radial direction sampling. The sampled directions (left) and the resulting roadmap (right) are shown. . . .	95
4.10	The test domain for interactive and benchmark testing of Dynamic PRM. . .	96
4.11	Path smoothing using the leader-follower approach.	100
4.12	An example of a specially handled collision check when the initial position is partially penetrating an obstacle. The collision check is handled in two segments.	101

4.13	A kinodynamically-limited search tree (left), and the corresponding simplified plan (right) for the UAV. The plan length is approximately 11km with waypoints every 100m. Obstacle clearance is shown as a translucent tunnel. . . .	102
4.14	An example of limiting Kinematics for the RRT extend operator	103
4.15	An example of a goal updating dynamically during search to try to maintain a tangent constraint. The original tangent, shown dotted (left), is updated as a the ERRT search tree expands (right).	104
4.16	Two examples of the fly-to-orbit capability of the UAV planner.	104
4.17	An occupancy grid and its distance transform.	106
4.18	QRIO on top of an autonomously reached step obstacle.	107
5.1	Drive layout for a CMDragons holonomic robot with four omni-directional wheels (left), and a picture of a partially assembled robot (right).	111
5.2	The model of $A(v)$ used in applying DSS to small-size soccer robots. F is the maximum acceleration and D is maximum deceleration. It is oriented by the current velocity v	113
5.3	Example environment shown in world space and velocity space. Note that this figure is hand-drawn and thus only approximate.	115
5.4	An example of an iteration of DSS with two agents. Each agent starts by assuming it will stop (a), and then each agent chooses an action (b)-(c), while making sure the action will allow a safe stop afterward. Finally, the actions are executed (d) and the agents can safely assume that stopping is a valid action.	121
5.5	Example velocity profile of two agents i and j . Each agent starts at a distinct velocity and executes a control acceleration for time C , and then comes to a stop using deceleration D . This defines three segments of relative motion, each with a constant acceleration.	127

5.6	An example situation showing n agents with $\Omega(n^2)$ overlapping trajectories.	128
5.7	The evaluation environment for the DSS algorithm.	129
5.8	Multiple robots navigating traversals in parallel. The outlined circles and lines extending from the robots represent the chosen command followed by a maximum rate stop.	130
5.9	Comparison of navigation with and without safety search. Safety search significantly decreases the metric of interpenetration depth multiplied by time of interpenetration.	132
5.10	Comparison of several margins under increasing vision error. The four different margins used are listed in the key, while increasing vision standard deviation is plotted against the collision metric of interpenetration depth multiplied by time of interpenetration.	133
5.11	Average execution time of safety search for each agent, as the total number of agents increases. For each robot count, 100 trials of the left-right traversal task were run. Both the raw data and means are shown.	135
5.12	A sequence captured while running DSS with four CMDragons robots. All agents completed their 2.8m traversal within 3.1 seconds.	135
5.13	A sequence captured while running DSS with four CMDragons robots and five small static obstacles. All agents completed their 2.8m traversal within 2.8 seconds.	136
6.1	Algorithm steps in RRT	144
6.2	Example Growth of RRT	145
6.3	RRT extensions with obstacles	146
6.4	RRT as a motion planner	147

6.5	Uniformly Sampled PRM Example	147
A.1	A video image and its YUV color classified result	156
A.2	A YUV histogram with a threshold defined	157
A.3	A 3D region of the color space represented as a combination of 1D binary functions.	159
A.4	An example of how runs are grouped into regions	161
A.5	The CMDragons'02 Robots. More recent robots use use the same marker pattern.	165
A.6	Aggregate error distribution for all samples including all three patch diameters. This is actual data collected from the vision system.	168
A.7	Cumulative distributions of absolute error. Note that patch size does not have a large effect on error. Along the direction of travel, the largest patch size decreases error somewhat, but does worse than the smallest patch size perpendicular to the direction of travel.	169
A.8	Examples of common tracking patterns from the RoboCup F180 environment.	170
A.9	Comparison of the positional and angular error of different patterns as the error of individual patches vary. For relatively small patch standard deviations, a linear relationship exists between the two, although the number of patches and layout of the pattern vary the factor.	175
A.10	2D scatter plots of adjacent readings for single patches (top) and for a full Butterfly pattern (bottom). Uncorrelated readings would show up as a circular 2D Gaussian cloud. Note the structure in the plot for a single patch, and the unstructured but non-circular distribution for the full pattern.	177

B.1	The overall system architecture for CMUnited/CMDragons. Thanks in particular to Brett Browning and Michael Bowling for their contributions to the overall team.	181
B.2	The general architecture of the CMDragons offboard control software. . . .	182
B.3	View of the robot drive system, kicker, and dribbler (left), and an assembled robot without a protective cover (right).	183
B.4	The main robot electronics board for CMDragons 2006.	184
B.5	Model of a CMDragons holonomic robot with four omni-directional wheels (left), and a picture of a partially assembled robot (right).	185
B.6	Our motion control approach uses trapezoidal velocity profiles. For the 2D case, the problem can be decomposed into two 1D problems, with the division of control effort balanced using binary search.	187
B.7	An example situation demonstrating the passing evaluation function inputs. The input values for evaluating a point p are the circular radius and subtended angle of the arcs a and b , and the angle between the center line segments of the two arcs. These are combined using simple weighted functions to achieve the desired behavior.	190
B.8	Two example situations passing situations are shown with, the passing evaluation metric sampled on a regular grid. The values are shown in grayscale where black is zero and the maximum value is white. Values below 0.5% of the maximum are not drawn. The same evaluation function is used in both the short and long pass situations, and the maximum value is indicated by the bold line extending from the supporting robot.	192
B.9	System model for one-touch ball control. The final velocity of the ball v_1 still contains a component of the initial velocity v_0 , rather than running parallel to the robot heading R_h	193

List of Tables

1.1	A generic motion planning and replanning algorithm	30
1.2	Guide to the thesis	38
2.1	A summary of the planning properties of several real robot domains	46
3.1	Algorithm for constructing and using a basic spatial hash table	53
3.2	Algorithm for building a spatial tree with per-node bounding boxes	56
3.3	Algorithm for checking a spatial tree with per-node bounding boxes	58
3.4	Algorithm for building a mask signal	60
3.5	Algorithm for querying a mask signal	61
3.6	Our method to evaluate a splitting plane for building a hierarchical bounding box representation for collision checking.	61
3.7	Algorithm for checking a swept query based on distance	64
4.1	The basic RRT planner stochastically expands its search tree to the goal or to a random state.	75
4.2	A basic implementation of a PRM planner	77

4.3	The code for a one-shot PRM planner	78
4.4	The extended RRT planner chooses stochastically between expanding its search tree to the goal, to a random state, or to a waypoint cache.	78
4.5	The parameter values for ERRT used in the benchmark experiment.	89
4.6	A comparison of ERRT with the visibility graph method across several 2D domains.	91
4.7	A comparison of performance of Dynamic PRM with various approaches for roadmap connection.	97
5.1	The Dynamic Window method for a single agent	116
5.2	A comparison of properties of Dynamic Window, explicit planning, and Dynamics Safety Search	117
5.3	The high level search routines for velocity-space safety search.	118
5.4	Robot-robot checking primitive for safety search.	122
5.5	The parabolic trajectory segment check.	123
6.1	Comparison of related planning algorithms	152
A.1	Performance of the CMVision low-level color vision library at various resolutions	163
A.2	The estimated standard deviations and 95% confidence intervals by patch size.	170
A.3	The number of uniquely identifiable patterns that can be detected using a certain number of colors (excluding the key patch and key patch color) . . .	172
B.1	Results of RoboCup small-size competitions for CMDragons from 2001-2006	195