The Data Market:

Policies for Decentralised Visual Localisation

Matthew Gadd
Keble College



Oxford Robotics Institute
Department of Engineering Science
University of Oxford

A thesis submitted for the degree of
Doctor of Philosophy

Trinity 2017

Matthew Gadd
Keble College

Doctor of Philosophy
October, 2017

The Data Market:
Policies for Decentralised Visual Localisation

Abstract

This thesis presents a *Data Market* for the dissemination of expertise in the form of locally accurate topometric maps which are agnostic to sensor modality and estimation strategies amongst arbitrarily large teams of autonomous vehicles with a common goal of localising well in a dynamic world.

Cooperation is at first centralised and allows for rapid construction of considerable maps by the conjoining of work uploaded by asynchronous processes before sharing expertise amongst the *fleet* by a selective downloading of mission-specific map contents. Embedding the centralised case within formalisations supporting distributed version control systems imposes obligatory commutation properties which ensure consistent integrity and eventual convergence in the state of each agent's map, allowing us to consider how any number of decentralised agents operate in concert via data sharing policies that are germane to the shared task of lifelong localisation. *Vendors* in the market intermittently initiate exchanges based on either mission parameters or a demand for localisation capability and choose trading partners with discrimination based on an internally evolving set of beliefs in each other's value as providers over a catalogue of distinct places of interest – which are treated as *products* for appraisal and purchase.

The system is implemented in more than 50 000 lines of C++ code and evaluated over hundreds of kilometres of monocular and stereo imagery from a comprehensive collection of warehouse, urban, and planetary analogue environments featuring diverse deviation in appearance due to textural, atmospheric, lighting, and structural dynamics and analysed over an exhaustive combination of the sensory records of the *Oxford RobotCar Dataset*.

Statement of Authorship

This thesis is submitted to the Department of Engineering Science, University of Oxford, in fulfilment of the requirements for the degree of *Doctor of Philosophy*. This thesis is entirely my own work and, except where otherwise stated, describes my own research.

Matthew Gadd, Keble College

Funding

Matthew Gadd is supported by the *FirstRand Laurie Dippenaar*, *Oppenheimer Memorial Trust*, and *Ian Palmer* scholarships, as well as the European Community's Seventh Framework Programme under grant agreement FP7-610603 (EUROPA2).

Dedication

Raja

bits, bots, bytes, and beats

gliding amongst the stars to

celestial chill

Acknowledgements

For his fierce imagination, patience, and sense of humour I would like to thank my supervisor, Professor Paul Newman. Likewise, my fellow students at the Oxford Robotics Institute – particularly Cori, Joffels, Zhuli, Boo, Bartie, Jeff, Amayo, Hugo, and Terry – who have allowed me to share in their brilliance and laughed with me along the way.

To all of the teachers, rascals, and imaginary friends who have appeared in recent years – Bwana, Lynn, Jack, Elmwood, Annabru, Max, Big Ferg, Edge, Marielle, Timmah, Leon, Annie, Jeanne, Shea, Libster, Hare, Ingo, Vincent, and Mabey – as well as all of the characters and each of the bricks that made Keble College a home and a playground.

Finally, to my parents and sister, who have constantly had my back.

Matthew Gadd

Friday 7th October, 2017

Contents

List of Figures **vii**

List of Tables **xi**

List of Algorithms **xii**

List of Abbreviations **xiii**

List of Symbols **xv**

1 Introduction **1**

1.1 Motivation 1

1.2 Guiding Questions 4

1.3 Contributions 6

1.4 Publications and Exploitation 7

1.5 Thesis Structure 9

2 Lifelong Vision-Based Navigation **13**

2.1 Large-Scale Localisation 14

2.1.1 Visual Odometry 15

2.1.2 Constrained Estimation 21
2.1.3 Place Recognition 24
2.2 Lifelong Navigation 28
2.2.1 Topometric Maps 28
2.2.2 Visual Teach-and-Repeat 30
2.2.3 Experience-Based Navigation 32
2.2.4 Predictive Approaches 34
2.2.5 Collaborative Localisation 36
2.2.6 Map Maintenance 38
2.3 Localisation Metrics 39
2.3.1 Accumulating Failures 39
2.3.2 N-Fold Cross-Validation 42
2.3.3 Randomised Trials 44
2.4 Implications on Fleet Operations 45
2.5 Summary 48

3 An Introduction to Version Control and Patch Theory 50
3.1 Basic Elements of Patch Theory 51
3.2 Repository Model 55
3.2.1 Working Copy 56
3.2.2 Repository Context 59
3.2.3 Encoding Map Updates 61
3.3 Branching Model 63
3.3.1 The Local Master Branch 63
3.3.2 Manipulating the Model 64

3.3.3 Conflict Resolution 71
3.4 Trading Workflow 72
3.4.1 Product Policies 72
3.4.2 Itemising Data Exchanges 76
3.5 Summary 80

4 A Version-Control View of Vision-Based Navigation 82

4.1 Graphs 83
4.1.1 Characteristics of Place and Motion 84
4.1.2 Graph Tools 89
4.2 Localisation as a Metric Graph Relationship 92
4.3 Confirming Representations 97
4.3.1 The Experience Graph 97
4.3.2 Stereo Localisation in the Experience Graph 100
4.4 Confirming Darcs Attributes 101
4.4.1 Incorporating Foreign Experience 102
4.4.2 Maintaining Coherency in Pairwise Exchanges 104
4.4.3 Maintaining Map Integrity Under Patch Updates . . . 108
4.5 Results 113
4.5.1 Commutation Policy Verification 114
4.5.2 Convergence Properties 117
4.5.3 Rejecting Graph Representation Disturbances 118
4.5.4 Commuting and Inducing Forgetfulness 120
4.6 Summary 123

5 Centralised and Distributed Visual Navigation 126

5.1 Trading Configurations 127

5.1.1 Centralised 128

5.1.2 Decentralised 129

5.2 Task Coordination 130

5.2.1 Processing Blocks 130

5.2.2 The Central Authority 135

5.2.3 Self-Organisation 138

5.3 Networking Protocol Suite 139

5.3.1 A Multirobot Finite-State Machine 140

5.3.2 Synchronisation Interfaces 141

5.3.3 Executing a Data Exchange 142

5.4 Results 144

5.4.1 Building Good Maps Quickly 145

5.4.2 Optimising Map Construction 146

5.4.3 Centralised Sharing of Expertise 148

5.4.4 Decentralised Sharing of Expertise 150

5.4.5 Scaling Marketplace Interactions 154

5.4.6 Mitigating Resource Issues 156

5.5 Summary 158

6 Trading in Lifelong, Multi-Agent Localisation 161

6.1 Targeting Content 162

6.1.1 Checking in Asynchronously 162

6.1.2 Correlated Checkouts 164

6.1.3 Throttling and Ranking by Value 170
6.1.4 Targeted Shopping Lists 172
6.2 Targeting Providers 174
6.2.1 Pricing Data Transactions 175
6.2.2 Tracking Trust in Vendors 177
6.2.3 Choosing Trades with Discrimination 180
6.3 Assigning Leading Value Providers 182
6.4 Results 184
6.4.1 Distributing Mission-Specific Maps 186
6.4.2 Valuing Data in the Market 189
6.4.3 Limiting Throughput by Value 190
6.4.4 Managing a Useful Shopping List 191
6.4.5 Catalogue Tool Caching 193
6.4.6 Targeting Valuable Interactions 196
6.4.7 Exploiting Trusted Vendors 197
6.4.8 Asymmetric Beliefs 198
6.4.9 Exploiting Local Expertise 202
6.5 Summary 203

7 Conclusion **206**

7.1 Summary 206
7.2 Future Work 208
7.3 Closing Remark 210

A Datasets **212**

Bibliography **234**

List of Figures

1.1 Stark as well as gradual change in appearance of a scene from Central Oxford over the course of one year. 2
1.2 Typical workflow for a mapping client as a participant in the DM. 4
1.3 Structure of this thesis. 10

2.1 An outline of a vision-based mapping and localisation pipeline. 14
2.2 Sample imagery showing interesting features. 17
2.3 Degrading motion estimation accuracy. 20
2.4 Motion inference with monocular and stereo techniques. 23
2.5 Visual similarity matrices. 26
2.6 A very simple T&R experiment illustrating the VT&R framework. 29
2.7 An overview of the key processing steps in a VT&R system. . 31
2.8 An implementation of monocular EBN localising over rich textures. 33
2.9 Cumulative failure distributions for maps built from indoor and outdoor textures. 42

3.1 Commuting square for generalised patches. 53
3.2 Topometric maps are implemented as graph structures stored within a database. 57
3.3 An agent's repository consisting of a swathe of representations. 60
3.4 Example experience map built using planar-induced homographies. 62
3.5 Mutual visualisation of the *history representation* for a *fleet* of robots. 64
3.6 Illustrating commutation of experience graph data from the perspective of experiences. 67
3.7 The *product policy*, $\wp$, expressed as a route network. 73

4.1 Camera coordinate system and projection model. 85
4.2 Localisation as a metric graph relationship. 92
4.3 Seeding pose estimation using metric searches. 94
4.4 An illustration of the topology expressing the *graph representation* of the *canonical DM client*. 97
4.5 A pairwise exchange of patches for the *canonical DM client*. . 102
4.6 Exemplar test cases for maintaining map integrity. 110
4.7 Generative simulator to verify the integrity of *graph representations* under the action of the *commutation policy*. 112
4.8 Certain and uncertain database growth under the action of mapping and trading. 114
4.9 Cumulative distributions of sampled simulation times for various *commutation policies* over fifty trials. 116
4.10 Steady-state database corruption time series. 118
4.11 Incremental database growth under the action of distinct strategies for commuting divergent content. 121
4.12 A selective consolidation of each agent's input using Γ_{darcs} deletes views from experiences and causes the EBN mapping module to record more information. 122

5.1 Illustration of communication topologies used in this thesis. . . 127
5.2 Generalised ensemble of processing blocks which are used to coordinate navigation tasks. 131
5.3 Centralised map assembly and sharing information amongst multiple robots. 135
5.4 State transitions for a robot entering into and participating in the DM. 140
5.5 Communication protocol for an exchange of patches and the associated *node representations* and *edge representations*. . . . 142
5.6 Building good *graph representations* quickly by devolution of map building tasks to subordinates. 145
5.7 Cumulative failure distributions for the average localisation performance of the held-out sets when subordinates process video sequences in various orders. 147
5.8 Centralised consolidation of maps between a pair of robots. . . 149
5.9 Decentralised cumulative localisation failures for BEVINGTON and BROAD. 150
5.10 Decentralised *distributed-query-all* cumulative localisation failures for ORI and TRINITY. 151
5.11 Scaling marketplace CPU and network activity for teams of increasing membership. 153

5.12 Trade-off in localisation performance and reduced disk load by the *distributed-query-one sales policy*. 155
5.13 Reducing CPU and network loads by the *distributed-query-one sales policy* in MAS of fixed membership. 157

6.1 Asynchronously checking in of chains or segments of frames that are connected linearly by ego-motion. 163
6.2 Discovering co-occurrence in path memory. 166
6.3 Choosing trading partners with discrimination. 179
6.4 Imposing organisational structure on the data as fed to agents for experimentation. 183
6.5 Cumulative failure distributions for correlated checkout techniques. 185
6.6 Cumulative database sizes and average time spent localising for correlated checkout techniques. 186
6.7 Cumulative failure distributions when embedding FAB-MAP or quality of ego-motion estimation within the *throttling tool* and *ranking tool* implementations. 188
6.8 Ranking throughput to the *canonical DM client* and throttling network *node representations* by using the *choice tool* as a value metric. 189
6.9 Tuning localisation outlook and pairwise correspondences by exploring the *product* space using peer-peer referrals. 192
6.10 Tuning CPU and network loads by endowing agents with a more nuanced understanding of the space in which they are operating. 194
6.11 Cumulative failure distributions for all decentralised *sales policies*. 198
6.12 Average bag of words cosine similarity from each dataset log to all others as assigned randomly or greedily to each MAS collaborator. 199
6.13 Aggregated tracked trust factors and corresponding error signals between the discriminative, decentralised *sales policies* and the unrestricted *distributed-query-all* approach. 200
6.14 Cumulative localisation failure distributions representing runtime instrumentation resolved according to the *product policy* over the RHODES and UNI-CLUB *product* spaces 201
6.15 Cumulative localisation failure distributions representing runtime instrumentation resolved according to the *product policy* over the HOLYWELL and TRINITY *product* spaces 203

A.1 Wireframe depictions of *Clearpath Husky A200*, *Pointgrey Firefly*, and *Neobotix MP-70* 213
A.2 Texture samples. 214
A.3 ARCQ14 and *Point Grey Grasshopper* mounted towards the front of the rover platform bed. 215
A.4 Exemplar martian analogue textures. 216
A.5 The BEGBROKE-24HR dataset. 217
A.6 Top-down and side-view schematics for *Nissan Leaf RobotCar* complete with sensor suite and frames of reference, taken from the *Oxford RobotCar Dataset* [Maddern et al., 2017] and available at `robotcar-dataset.robots.ox.ac.uk`. 218
A.7 GPS traces and an example *product* space over the *Oxford RobotCar Dataset*. 219
A.8 Typical appearance change within BEVINGTON, BROAD, BLACKHALL, and OBSERVATORY *product* categories. 221

List of Tables

4.1 Corruption rates expressed as the fraction of data exchange events at which the *graph representation* is within an acceptable threshold of perfect. 119

5.1 Decentralised *distributed-query-all* area-under-curve (AUC) results for MUSEUM, PARKS, PHYSICS and PITT-RIVERS. 152
5.2 Decentralised *distributed-query-all* percentage-not-localised (PNL) rates for BLAVATNICK and GLOUCESTER. 152
5.3 Decentralised *distributed-query-one* end-of-simulation node counts for NORTH-STORE and PHOENETICS. 158

6.1 Improved accumulated network load as a result of demarcating data in the DM by use of the *product policy*. 195
6.2 Improved accumulated CPU load as a result of demarcating data in the DM by use of the *product policy*. 195
6.3 Using an appearance-based metric within the *choice tool* implementation to tune the number accumulated matches over all trades. 196
6.4 Cumulative matches discovered for all decentralised *sales policies*. 199

A.1 Exemplar demarcation of the *Oxford RobotCar Dataset*. 220

List of Algorithms

1 Discovering chains or segments of frames that are connected linearly by ego-motion. 75
2 Pseudo-code for the *commutation policy*, using the *canonical DM client* to disregard redundant patch content. 105
3 Populating the path count matrix. 168

List of Abbreviations

6DoF ..... Six degree-of-freedom
AUC ..... Area-under-curve
BRISK ..... Binary robust invariant scalable keypoints
API ..... Application Programming Interface
BoW ..... Bag of Words
BRIEF ..... Binary Robust Independent Elementary Features
CAV ..... Connected and Autonomous Vehicle
CL ..... Collaborative localisation
CPU ..... Central Processing Unit
Darcs ..... Darcs Advanced Revision Control System
DM ..... *Data Market*
EBN ..... Experience-Based Navigation
EKF ..... Extended Kalman Filter
ESA ..... European Space Agency
FAB-MAP ..... Probabilistic Localization and Mapping in the Space of Appearance
FAST ..... Features from Accelerated Segment Test
FOV ..... Field of view
FPSBA ..... First-price sealed-bid auction
FREAK ..... Fast Retina Keypoint
FSM ..... Finite-state machine
GPS ..... Global Positioning System
IDF ..... Inverse Document Frequncy
MAS ..... Multi-Agent System
MOOS ..... Mission-oriented Operating Suite
MURFI ..... Mars Utah Rover Field Investigation
NED ..... North-East-Down
poset ..... Partially Ordered Set
RAM ..... Random Access Memory
RANSAC ..... Random Sample Consensus
SIFT ..... Scale-invariant Feature Transform

SLAM ..... Simultaneous Localisation and Mapping
SQL ..... Structured Query Language
SURF ..... Speeded-up Robust Features
SUSAN ..... Smallest Univalue Segment Assimilating Nucleus
SVD ..... Singular-value Decomposition
SVN ..... Subversion
TCP ..... Transmission Control Protocol
T&R ..... Teach-and-repeat
UUID ..... Universally Unique Identifier
VCS ..... Version-Control System
VO ..... Visual Odometry
VSM ..... Visual Similarity Matrix
VT&R ..... Visual teach-and-repeat
GN ..... Gauss-Newton
GRIEF ..... Generated BRIEF
LHS ..... Left-hand-side
LM ..... Levenberg-Marquardt
LSE ..... Least squares estimation
OCC ..... Optimistic Concurrency Control
ORB ..... Oriented FAST and Rotated BRIEF
P3P ..... Perspective from three points
PNL ..... Percentage of frames not localised
PR ..... Precision-recall
RHS ..... Right-hand-side
SVD ..... Singular-value decomposition
TF-IDF ..... Term Frequncy - Inverse Document Frequency
TF ..... Term Frequency
UCB ..... Upper confidence bound
WiFi ..... Wireless Fidelity

List of Symbols

Images

$\mathbf{X}$ . . . . . 3D point coordinates
$\mathbf{x}$ . . . . . Pixel coordinates
$\mathcal{I}$ . . . . . Image
$\bar{\mathbf{x}}$ . . . . Homogeneous pixel coordinates
$\bar{\mathbf{X}}$ . . . Homogeneous point coordinates

Features

$\mathbf{d}$ . . . . . Feature descriptor
$\mathbf{w}$ . . . . . Visual word
$\mathbf{b}$ . . . . . Bag of Words
$\mathbf{V}$ . . . . . Visual Similarity Matrix
$\mathbf{v}$ . . . . . Visual Similarity Vector

Simulation

k . . . . . Iteration, or time step
$t(k) = t_k$ . . UNIX timestamp at time step
$\theta_{i,k}$ . . . . . Flow state type
$\psi_{i,k}$ . . . . . Semaphore primitive
e . . . . . Epoch
$\mathbf{e}$ . . . . . Epochs
t_s . . . . . Time for sample
e_s . . . . . Epoch for sample

Fleet

r . . . . . Robot identifier
$\mathfrak{R}$ . . . . . Team identifiers
$|\mathfrak{R}|$ . . . . . Fleet size
$\mathfrak{R}_{\text{SELLERS}}$ . . . . . Requested agents
$r_{\circledcirc}$ . . . . . The central authority

Camera

$\mathbf{K}$ . . . . . Camera intrinsic calibration matrix
$\mathbf{P}$ . . . . . Projection matrix
$\mathbf{F}$ . . . . . Fundamental matrix
h . . . . . Camera height

Pose

$\mathcal{F}$ . . . . . Frame of reference
$\mathcal{F}_{\text{LIVE}}$ . . . . . Query frame of reference
$\mathbf{T}$ . . . . . Rigid-body transformation
$\mathbf{R}$ . . . . . Rotation
$\mathbf{R}_x(\alpha)$ . . . . . Pure rolling rotation
$\mathbf{R}_y(\beta)$ . . . . . Pure pitching rotation
$\mathbf{R}_z(\phi)$ . . . . . Pure yawing rotation
$\mathbf{t}$ . . . . . Translation
$\mathbf{H}$ . . . . . Homography
$\mathbf{q}$ . . . . . Quaternion

Localisation

f . . . . . Localisation failure length
$\mathbf{f}$ . . . . . Localisation failure lengths
d . . . . . Distance travelled between localisation failures
$\mathbf{d}$ . . . . . Localised distance
D . . . . . Total distance travelled

Parameters

ψ . . . . . RANSAC inlier threshold
$d_{\max}$ . . . . . Graph search extent
$\eta_{\mathbf{t}}$ . . . Translation threshold, keyframe addition
$\eta_{\mathbf{R}}$ . . . . . Rotation threshold, keyframe addition
N . . . . Minimum number of localisers
$\varkappa$ . . . . . Average simulated patch size (nodes)
Θ . . . Standard deviation in simulated patch size
$\mathfrak{S}$ . . . . . Sample size
$\mathfrak{T}$ . . . . . Throttle size
$\mathfrak{L}$ . . . . . Rank size
t_w . . . . . Time window
t_s . . . . . Average sampe time
λ . . . . . Best paths percentage
ϵ . . . . . Trial exploration ratio
T . . . . . Number of montecarlo trials
N . . . . . Cross-validation factor
R . . . . . Minimum RANSAC trials

Datasets

ℓ . . . . . Dataset log
$\mathcal{D}_i$ . . . . . Dataset log group
$\mathcal{D}$ . . . . . All dataset logs
$\mathcal{W}$ . . . . . The world, or environment

Database

v . . . . . Graph node
e . . . . . Graph edge
V . . . . . Graph nodes
E . . . . . Graph edges

Repository

p . . . . . Patch element
g . . . . . Graph element
ι . . . . . Patch element action
æ . . . . . Patch element adjacency list

States

$\overset{o}{\Xi}$ . . . . . Common input state
$\overset{*}{\Xi}$ . . . . . Pristine state
$\overset{p}{\Xi}$ . . . . . Protected state
$\overset{c}{\Xi}$ . . . . . Common output state
$\overset{u}{\Xi}$ . . . . . Virtual diff state
$\overset{*}{\Xi}$ . . . . . Pristine state

Patches

$\overset{a}{\Xi}\mathbf{P}\overset{c}{\Xi}$ . . . . . LHS convergent patch
$\overset{b}{\Xi}\mathbf{P}\overset{c}{\Xi}$ . . . . . RHS convergent patch
$\overset{a}{\Xi}\mathbf{P}\overset{u}{\Xi}$ . . Virtual LHS convergent patch
$\overset{b}{\Xi}\mathbf{P}\overset{u}{\Xi}$ . . Virtual RHS convergent patch
$\overset{a}{\Xi}\mathbf{P}\overset{p}{\Xi}$ Protected LHS convergent patch
$\overset{b}{\Xi}\mathbf{P}\overset{p}{\Xi}$ Protected RHS convergent patch
$\overset{a}{\Xi}\mathbf{P}\overset{s}{\Xi}$ . . . . . Sampled patch
$\overset{b}{\Xi}\mathbf{P}\overset{t}{\Xi}$ . . . . . Throttled patch

$^{a}_{\Xi}\mathbf{P}^{r}_{\Xi}$ . . . . . Ranked patch
$^{t}_{\Xi}\mathbf{P}^{b}_{\Xi}$ . . . . . Upload patch
$^{u}_{\Xi}\mathbf{P}^{r}_{\Xi}$ . . . . . Download patch

Products

ρ . . . . . Product
$\mho$ . . . . . Product catalogue
ρ_{LIVE} . . . . . Placement product
ρ_{QUERY} . . . . . Query product
ρ_{PREVIOUS} . . . . . Previous product
ρ_{NEXT} . . . . . Upcoming product
$\rho_{\text{RECOMMEND}}$ . . . Recommended product

Policies

Γ . . . . . *commutation policy*
$\mathbb{E}$ . . . . . *rollback tool*
$\wp$ . . . . . *product policy*
Ω . . . . . *sales policy*
$\overline{\wedge}$ . . . . . *upload policy*
$\veebar$ . . . . . *download policy*
ζ . . . . . *shopping policy*

Tools

φ . . . . . *placement tool*
$\mathcal{L}$ . . . . . *category tool*
$\mathcal{S}$ . . . . . *stock tool*
Ψ . . . . . *search tool*
γ . . . . . *choice tool*
$\mathbb{S}$ . . . . . *sampling tool*
$\mathbb{L}$ . . . . . *ranking tool*
$\mathbb{T}$ . . . . . *throttling tool*
Λ . . . . . *pricing tool*
$\hbar$ . . . . . *heatmap tool*
$\exists$ . . . . . *patching tool*

Representations

$\mathcal{G}$ . . . Database, *graph representation*, experience graph
Υ . . . . . Node representation
ξ . . . . . Edge representation
$\mathcal{R}$ . . . . . Repository
$\mathcal{H}$ . . . . . History
Ξ . . . . . State
$\mathbf{P}$ . . . . . Patch
$\mathfrak{f}_{i,k}$ . . . . . *flow representation*

Packets

$\mathfrak{C}$ . . . . . Unique channel identifier
ξ_{6DoF} . . . . . 6DoF ego-motion matrix
$\xi_{\text{translation}}$ . . . . . 3D translation vector
Υ_{BRIEF} . . . . . BRIEF binary pattern
Υ_{FAST} . . . . . FAST corner location
Υ_{image} . . Stereo camera image frames
Υ_{tod} . . . . . Image capture time
Υ_{SURF} . . . . . SURF feature vectors
Υ_{BoW} . . . . . Global image descriptor
Υ_{calib} . . . . . Calibration parameters
Υ_{paths} Path IDs associated to a node

Nodes

v_{LIVE} . . . . . Query node
V_{SEEDS} . . . . . Topological seeds
V_{RADIATE} . . . . . Candidate nodes for localisation
v_{*} . . . . . Localised node

Edges

E_{RADIATE} . . . . . Edges radiated from topological seeds

Commute

$\Gamma_{\cup}$ ..... Union *commutation policy*
$\mathcal{M}$ ..... Pairwise matches
Γ_{darcs} ..... Darcs *commutation policy*
v_{+} ..... Node chosen
v_{-} ..... Node disregarded
$v_{?}$ ..... *choice tool* indecision
p_{*} ..... Localised patch element
g_{*} ..... Localised graph element

Verification

$\mathcal{C}_i$ ..... Test configuration
$\mathcal{C}$ ..... Test configurations
$\mathcal{T}_j$ ..... Test case
$\mathcal{T}$ ..... Test battery

Communication Networks

$\Omega_{\text{dedicated}}$ ..... Dedicated requests
Ω_{central} ..... Centralised requests
Ω_{dqa} .. Distributed-query-all requests
$\Omega_{\text{bandit-explore}}$ ..... Random request
$\Omega_{\text{bandit-exploit}}$ .... Exploitative request
$\Omega_{\text{bandit-explore-exploit}}$ ..... Explorative-exploitative request

Path memory

$\mathcal{P}$ ..... Path UUID
$\mathcal{P}$ ..... Path memory UUIDs
$\mathcal{P}_s$ ..... Selected paths
$\mathbf{M}$ ..... Path memory matrix
$\mathbf{C}$ ..... Path count matrix
$\mathbf{W}$ ..... Path weight matrix

Shopping lists

ζ_{CURRENT} ..... Current
ζ_{WINDOW} ..... Window
$\zeta_{\text{RECOMMEND}}$ ..... Recommend

Pricing

κ ..... Cost
δ ..... Reward
$\eth$ ..... Price
$\check{\kappa}$ ..... Tender cost
$\check{\delta}$ ..... Tender reward
$\check{\eth}$ ..... Tender price
$\top$ ..... Tender offers
μ ..... Empirical mean in price
σ ... Empirical standard-deviation in price

Chapter 1

Introduction

1.1 Motivation

Concerted action amongst a *fleet* of robots is a tangibly effective strategy for executing a common set of tasks (e.g. mapping, exploration, target tracking, path planning, and surveillance), particularly under realistic considerations of resource limitations (e.g. bandwidth, computational, and storage).

Retrofitting indoor, outdoor, or urban environments with infrastructure to accurately locate and position a *team* of robots may be undesirable in terms of both cost and flexibility. Indeed, in many contexts a Global Positioning System (GPS) offers intolerable imprecision and intermittent connectivity. Cameras, however, are low-cost, information-rich devices which robots can use to capture the geometry and appearance of a scene both unique to a particular viewpoint and at a resolution capable of expressing small changes in the world.

However, consider the montage shown in Figure 1.1; the world is subject



Figure 1.1: These images, more samples of which are available in Appendix A.2.2, are taken from the same location in Central Oxford over the course of one year. Variability in appearance (largely due to illumination in this example, but including adverse weather conditions brought about by rain) impedes reliable localisation for just one robot, and will scale poorly in complexity across the lifetime of operation of a *fleet* of robots equipped with heterogeneously configured but complementary sensor suites.

to both regular and sporadic modes of appearance change. Seamlessly dealing with stark scene change is difficult and agents acting in isolation or in tandem will need to deal with marked variation of scene appearance and structure over time.

Indeed, while scene change is often "experienced" at the level of individual agents, there is a need and great boon in intelligently sharing individual experiences across a collaborative *team* to improve fleetwide localisation performance as well as to avoid the ceaseless accumulation of redundant data. This will be particularly aggravated by the totality of experiences over the lifetime operation of a *fleet* of vehicles, where the question of sharing independently gathered experience data has not been adequately addressed, most simply manifest in the lack of an intuitive guide for the optimum consolidation of all sensory input available. Clearly, some management policies are required.

A reusable software module analogy Figure 1.2 shows the typical workflow for multi-agent navigation as presented in this thesis. Here, agents are analogous to software developers using version control systems (VCS) to simultaneously add to and improve upon a shared coding project, as well as choosing when to share those changes with the rest of – or some part of – the team. In enterprise software management, when two developers make conflicting edits to the same line of a file, the VCS requests human assistance in deciding what to do. In this thesis, however, changes to the versioned data are instigated by instances of an autonomous mapping and localisation process, which the VCS treats as the conflict resolution mechanism.

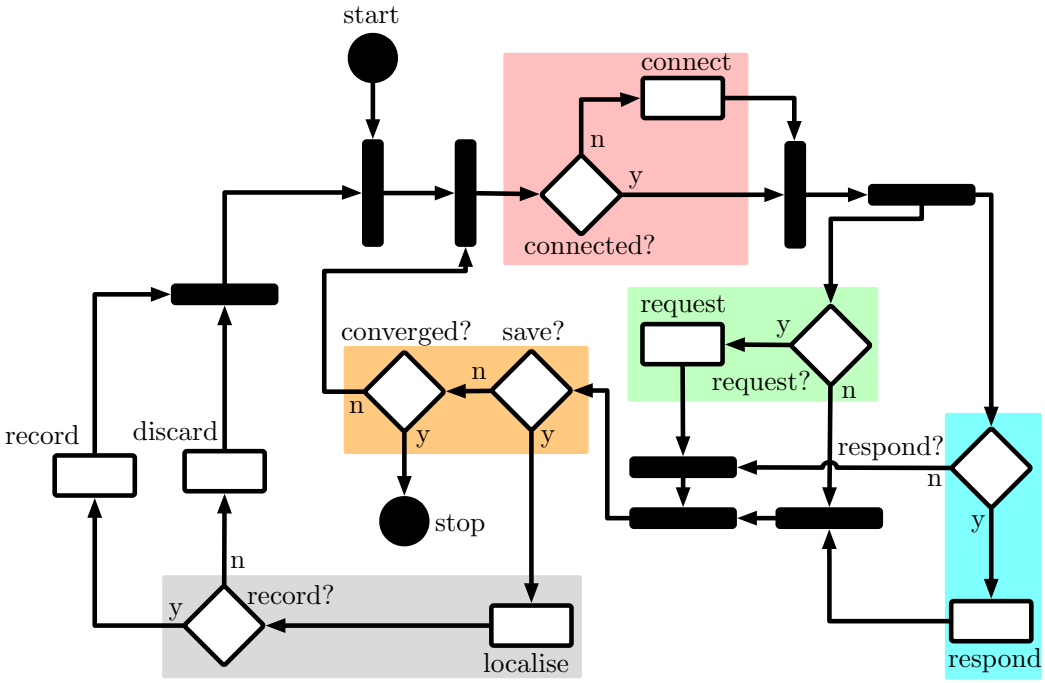


Figure 1.2: The typical workflow for a mapping client as a participant in the DM. While an agent is still in possession of raw sensor logs for processing, it first attempts to localise and discards said imagery if it is sufficiently explained by the current state of the map (grey shaded area). When the raw sensor log store on each agent is depleted, *team* members can optionally continue to perform exchanges until a convergence criterion is reached (orange shaded area). This workflow must be expanded to include the facility to initiate and respond to requests (green and blue shaded areas respectively).

1.2 Guiding Questions

This thesis is concerned with underpinning the long-term operational integrity of *fleet*s of vehicles using visual localisation. In particular, we explore stratagems for vehicles – considered as independent agents – to share, update, and leverage each others' visual experiences in a mutually beneficial way, and consider the consequential challenges in data management and exploitation that arise over the lifetime of operation of *fleet*s of autonomous vehicles.

In Sections 2.3 and 2.4 we will examine at a systems level the typical performance of an isolated (single-agent) state-of-the-art lifelong visual localisation system and use this insight to motivate several theoretical and practical questions which must be answered before distributing mapping responsibilities to a co-operative team, including:

1. Is there a natural formalisation of graph-based visual localisation systems which can provide reasonable guarantees that the team will always be able to consolidate divergent representations of the same environment, even under conditions of irregular or unreliable peer-peer connectivity?

2. How can we best leverage this formalisation to ensure scalable operation of the team over large scales and extended operational periods, in terms of CPU, disk, and network limitations?

3. Can we embed the various subroutines inherent to lifelong navigation suites of this kind through this formalisation in totally centralised or decentralised teams, and as such engage in targeted pairwise interactions?

4. If so, which aspects of these navigation suites, when propagated through the system, are most exploitable when targeting interactions towards improved localisation in a simple trading application?

We refer to the framework which emerges from this line of questioning as the *Data Market* (DM) and present it as a set of protocols and guiding principles for mapping clients to both contribute to in the form of gathered

navigation data, as well as leverage to promote their own localisation performance by exploiting the content input by locally expert *team* members.

1.3 Contributions

This thesis manifests itself as six contributions in the field of distributed, visual localisation:

1. Posing and modelling decentralised localisation as an instance of distributed version control: abstracting data fusion as a commutative operation which guarantees both synchronicity in representation amongst the *team* and the avoidance of map corruption.

2. An instantiation of the DM as a *central authority* through which any number of robots may operate in concert via data sharing policies that are suited to the shared task of lifelong localisation within a variable environment: rapidly constructing considerable maps by the conjoining of work distributed to asynchronous processes and sharing expertise amongst the *team* by a targeted administration of mission-specific map contents.

3. Abstracted protocols in the form of interfaces through which agents intermittently initiate trades based on a realtime demand for localisation capability, as well as a means for brokering exchanges navigation expertise with guaranteed synchronisation for exactly repeatable simulations of arbitrarily large *fleet*s with varied interests and configurations.

4. A candidate set of policies which agents use as value strategies to explore the *Data Market*, based on an evolving set of beliefs in the likely boon after making a purchase from another identifiable member of the *team*.

5. Efficient and scalable transformations of the *fleet*'s operational envelope via a demarcation of the data that can be collected within it over the lifetime of operation of the *fleet* as a "catalogue" of commodities which each robot can explore and contribute to.

Here, we phrase the ideas by using a set of representations which dictate a universally applicable graph-based framework for the navigation of robots equipped with a variety of sensors. This foregoes the need for a metrically accurate representation of the world, and allows for the "stitching" together of different experiences of adjacent areas into a large network.

1.4 Publications and Exploitation

This thesis leaves in its wake a publication trail which began with a novel instantiation of the Experience-Based Navigation (EBN) framework for small robotic platforms operating on relatively flat, highly textured surfaces in [Gadd and Newman, 2015]:

1. **M. Gadd** and P. Newman, "A Framework for Infrastructure-Free Warehouse Navigation," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Seattle, WA, USA, 2015.

This work was later demonstrated in partnership with the *European Space Agency* (ESA) within a planetary-exploration scenario [Yeomans et al., 2017] and disseminated publicly in [Balme et al., 2017]:

2. B. Yeomans, H. Porav, **M. Gadd**, D. Barnes, J. Dequaire, T. Wilcox, S. Kyberd, S. Venn, and P. Newman, "MURFI 2016 - From Cars to Mars: Applying Autonomous Vehicle Navigation Methods to a Space Rover Mission," in *14th Symposium on Advanced Space Technologies in Robotics and Automation (ASTRA)*, Leiden, The Netherlands, 2017.

Publications 1 and 2 are especially relevant to the initial discussion in Chapter 2, which provides background to lifelong visual navigation concepts, and to the small-to-medium scale experiments in Chapters 2 and 3 which are presented as discursive examples to bring attention to the various challenges discussed in Section 1.2 above.

The novel phrasing of version control as a metaphor for lifelong mapping and localisation was first introduced to the robotics community in [Gadd and Newman, 2016] while the set-theoretic formalisation was used in [Gadd and Newman, 2018] to design the totally distributed framework which we approach towards the end of this thesis:

3. **M. Gadd** and P. Newman, "Checkout My Map: Version Control for Fleetwide Visual Localisation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Daejeon, South Korea, 2016.

4. **M. Gadd** and P. Newman, "The Data Market: Policies for Decentralised Visual Localisation," *ArXiv e-prints*, 2018.

These publications used distinct trading configurations in applying the concepts presented in this thesis, and as such the results presented in both Chapters 5 pertaining to centralised and decentralised teams of agents and 6 are drawn from Publication 3 and Publication 4 respectively.

This thesis has begun, at the time of writing, to be exploited heavily within Innovate UK's DRIVEN consortium, which seeks to address challenges in insurance modelling, risk profiling, and data security which will be encountered by a fleet of fully autonomous vehicles being deployed in urban areas and on motorways, and will culminate in an end-to-end Connected and Autonomous Vehicle (CAV) trial from London to Oxford[1].

1.5 Thesis Structure

This thesis is structured as shown in Figure 1.3. Ultimately, the fully distributed *Data Market* requires no centralised processing or storage, but we have depicted it with a pair of prerequisites: "MAP" (the format of content being shared through the DM) and "API" (interfaces and strategies for such distribution).

Chapter outline Indeed, the concepts herein are applied to the visual localisation problem, and as such we introduce generally in Chapter 2 the background knowledge necessary to formulate a DM content category which encompasses topometric maps, with a particular and useful instance of a lifelong navigation suite detailed further in Chapter 4, which constitutes the

[1] `http://drivenby.ai`

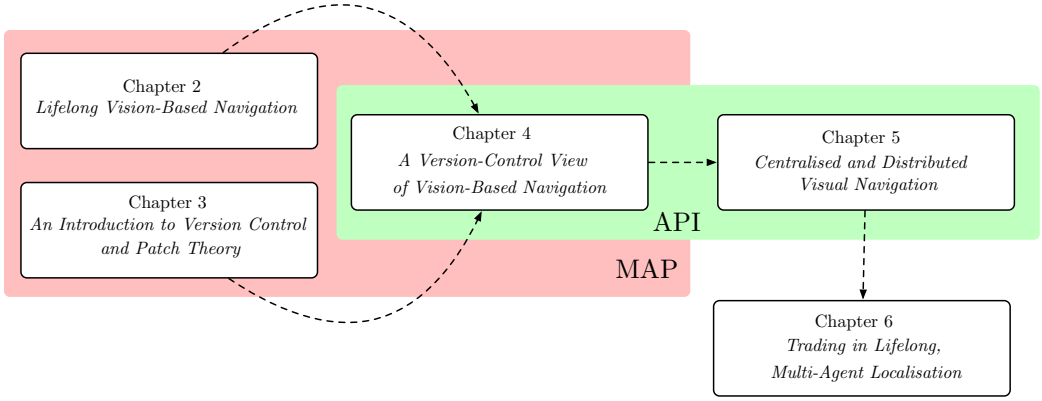


Figure 1.3: Effective access to the DM requires both a map ("MAP") and protocols ("API") for consistent versioning amongst all contributing agents, both of which require the common parlance of a set of representations, where this nomenclature has been borrowed from [Cieslewski et al., 2015].

canonical DM client employed throughout this work.

Chapter 2 concludes with a demonstrative discussion of the are myriad situational, resource, and design related choices regarding sharing and leveraging such navigation expertise and their implications on fleet operations. In recognition of this, we suggest two extremes in choice, and use Chapter 5 to introduce the DM first as a *central authority* for accumulating and dispensing mapping data and then in the distributed case as a robust set of interfaces which scales gracefully with mission, team, and environmental parameters.

Devolution of market strategies to individual agents in this way requires a formalisation of the experience mapping problem as an instance of distributed version control, which is undertaken in Chapter 4 and which relies heavily on the versioning notation introduced in Chapter 3.

In Chapter 6 we further explore strategies that *vendors* can employ to act discriminatively and effectively within the DM by applying the combined system to a trading scenario which targets meaningful pairwise interactions

and improved localisation over the course of many repeated sorties to an environment which exhibits high variability in appearance.

How to read this document This thesis makes use of several narrative devices which are useful for unifying the concepts which agents participating in the DM must adhere to and can leverage, including:

1. **Definition**: Incidental statements which are useful to isolate and are repeatedly referred to in later dependent text.

2. **Example**: Small-scale examples are used to develop the background knowledge for lifelong navigation, drawing on specimens which are detailed in Appendix A.1.

3. **Representation**: Abstracted formats for the navigation data to be shared across the *fleet*.

4. **Policy**: Strategies for both single-agent navigation as well as multi-agent collaboration have generalised input-output interfaces, where specific instantiations are proposed and investigated.

5. **Tool**: Algorithms, types, and verification strategies which are indispensable to understanding and implementing the policies and how they manipulate representations. Here, take as an example the *commutation policy* to be discussed in Section 3.3.2, which is used to modulate damaging effects of the *patching tool* as it acts upon the *graph representation*.

The notation and abbreviation arising from these assistive templates are summarised in the preamble material as a "List of Symbols" and a "List of Abbreviations" for the reader's use. Here, as per the example in item 5 above, the *commutation policy* (Γ) is expressed through the *patching tool* ($\exists$) in terms of the nodes and edges added to or removed from the *graph representation* ($\mathcal{G}$).

Throughout this thesis, experimental outcomes are expressed in terms of the performance metrics discussed in Chapter 2 and measured against a swathe of datasets which can be viewed in Appendix A – some data from which are also used as motivating examples in Chapters 2 to 4.

Ultimately, Chapter 7 concludes with a summary of this thesis, its contributions, and aspects of this work which are ongoing in the research community, industry, and the public domain.

Chapter 2

Lifelong Vision-Based Navigation

Once more using Figure 1.1 as a motivating example, the presence of complex modes of change in the environment's appearance (e.g. weather, illumination, or structure) makes visual localisation a challenging problem to solve. This chapter serves as an overview of concepts important to large-scale, lifelong navigation where we engage in – as motivating examples to develop these concepts – a few illustrative case-studies from the small-scale, indoor and outdoor datasets detailed in Appendix A.1, which were used in early development [Gadd and Newman, 2015] of lifelong navigation techniques for this thesis. To begin, we present a brief overview of the prerequisite estimation techniques and representations which support these systems fundamentally, as illustrated in Figure 2.1.

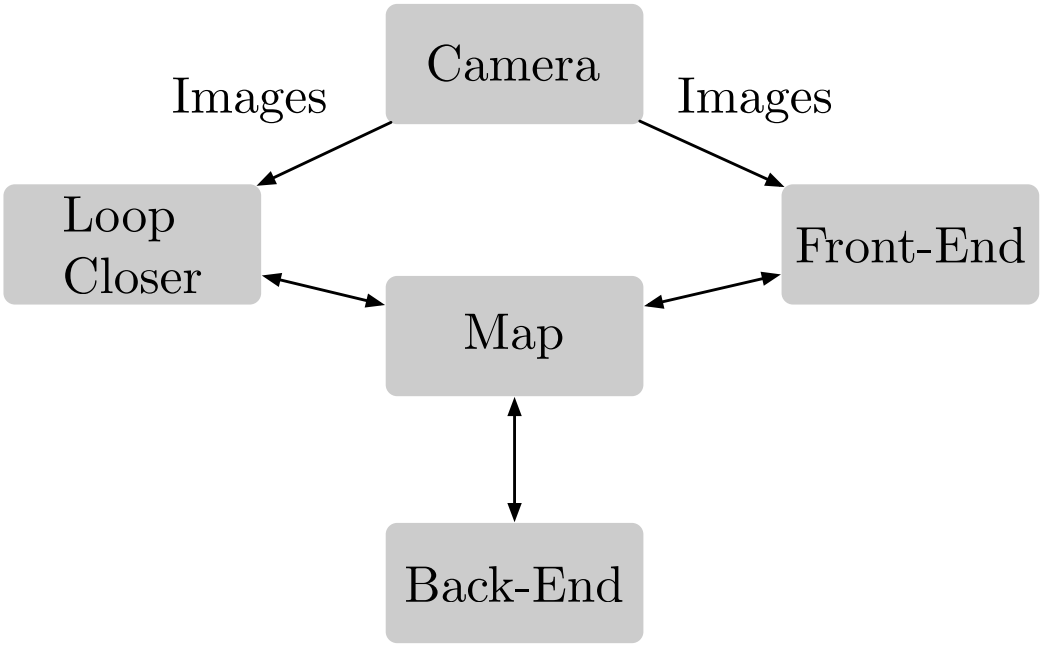


Figure 2.1: An outline of a vision-based mapping and localisation pipeline, adapted from [Churchill, 2012]. Images are passed to a front-end for local metric pose estimation (Section 2.1.1 and Section 2.1.2) and to a place recognition module (Section 2.1.3) for loop closure detection, both of which interface with a map (Section 2.2.1). While not discussed in this thesis, a back-end to the pipeline may continuously optimise the map.

2.1 Large-Scale Localisation

In outdoor environments, autonomous vehicles are required to deal with enduring (seasonal, weather, etc.) as well as ephemeral (obstacles, lighting, etc.) modes of appearance change. In this thesis, we use the following definition of the world or environment (terms which we will use interchangeably):

Definition 2.1.1. *Agents in the DM share an overlapping operational envelope, $\mathcal{W}$, the layout of which they are trying to capture through building maps in which they can reliably localise.*

Ephemeral changes, such as moving people, can be treated as a source

of noise and appropriately filtered [Konolige and Bowman, 2009], while enduring changes, such as the changing of the seasons, must be treated as an interruption to the relevance of an agent's representation of the world over longer time scales [Churchill, 2012].

Before addressing scaling *fleet* operations to lifetimes of operation in Section 2.2, we use the following sections to provide an overview of how robots can use visual sensors alone to recover motion, perform mapping, and localise within constructed representations in realtime.

The following sections use concepts of rigid-body transformations and basic image processing (for example feature matching) that are defined more fully in Chapter 4. In this Chapter we only require the reader to be aware of their use in higher level functional concepts, and not their detailed operation and construction - we delay the latter to the start of Chapter 4.

2.1.1 Visual Odometry

Visual Odometry (VO) [Nistér et al., 2004] identifies and associates visual clues to estimate the full 6DoF motion of a camera through free space as a sequence of incremental transformations, $\mathbf{T}_{k-1,k}$, between consecutive video images, $\mathcal{I}_{k-1}$ and $\mathcal{I}_k$, such that

$$\mathbf{T}_{k-1,k} = \begin{bmatrix} \mathbf{R}_{k-1,k} & \mathbf{t}_{k-1,k} \\ \mathbf{0} & 1 \end{bmatrix} \tag{2.1}$$

consisting of a rotational and translational component – more properties for which will be discussed in Section 4.3.1 below in the specific context of

a map built with stereo techniques. In contrast to wheel odometry, which counts wheel revolutions as the robot moves through the world, this is done without access to a motion model and is robust to unforeseen environmental factors which may induce wheel slip. A comprehensive investigation of VO techniques is available in [Fraundorfer and Scaramuzza, 2012], but we consider a few important aspects here.

Salient keypoint detection VO requires a robust method for identifying distinctive and efficiently retrievable features for landmarks, such as FAST corners [Rosten and Drummond, 2006], which are computationally less intensive than detectors such as Harris [Harris and Stephens, 1988] or SUSAN [Smith and Brady, 1997], and are better suited for use in realtime applications. Figure 2.2 shows two sets of keypoints where some (e.g. the red corners detected in the sky region of the left frame) are clearly more salient than others.

Baseline matching For monocular odometry, as will be discussed in Section 2.1.2, strong assertions of the locally planar nature [Forster et al., 2014] of the scene constrains the search for keypoint correspondence significantly. However, as to be detailed further in Section 4.1.1 when instantiating the *canonical DM client*, stereo odometry estimation proceeds by searching for data associations between reference keypoints in the left and right images of a stereo frame, and is constrained along a single column in each image by rectified, parallel, epipolar lines.

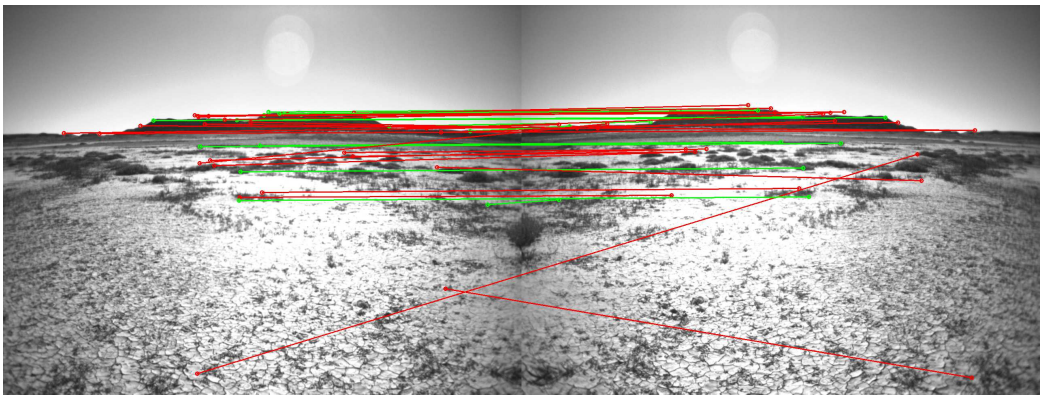


Figure 2.2: Sample imagery taken from Appendix A.1.2 showing interesting features (red and green, detected and described by SURF) on two consecutive images captured by an ARCQ14 space rover as it moves. Frame-to-frame matching using FLANN [Muja and Lowe, 2009] followed by a RANSAC-based homography estimation provides the distinction between inliers (green) and outliers (red) in this example.

Inverse disparity triangulation Since the disparity of the observations, stereo baseline, and intrinsic calibration of the cameras are known, the position of 3D landmarks can be triangulated. The camera's depth perception at distance is dependent on the maximum image resolution, camera focal length, and the stereo baseline. Loosely, depth becomes more uncertain as the disparity shrinks. This is an important consideration for scenery as shown in Figure 2.2, where the majority of the salient keypoints are detected at range (on or near the horizon).

Landmark description Robots do not revisit locations with exactly the same attitude and as such identifying landmarks in farflung pairs of stereo frames requires a quantised representation which is robust to changes in viewpoint and scale, such as the Scale-invariant Feature Transform (SIFT) [Lowe, 2004] or Speeded-up Robust Features (SURF) [Bay et al., 2006] (sam-

ples of which are shown in Figure 2.2).

However, incremental pose estimation for robots moving at a moderate speed allows for the use of more computationally efficient binary descriptors such as Binary Robust Independent Elementary Features (BRIEF) [Calonder et al., 2010]. Binary encodings of this kind have become popular due to their low computational and lenient data requirements, and include, amongst others:

1. Oriented FAST and Rotated BRIEF (ORB) [Rublee et al., 2011], which uses a pyramid to produce multiscale-features,

2. Binary robust invariant scalable keypoints (BRISK), which employs a hand-crafted sampling pattern described in [Leutenegger et al., 2011] as composed of concentric rings, and

3. Fast Retina Keypoint (FREAK) [Alahi et al., 2012], which uses a retinal sampling grid smoothed with a Gaussian kernel.

A Generated BRIEF (GRIEF) feature descriptor trained with evolutionary algorithms is proposed in [Krajník et al., 2017] which leverages binary descriptors of this kind for robustness to long-term, natural appearance change.

2D feature association Once identified, landmark descriptors must be associated with each other by a similarity metric, where a Euclidean distance or cosine similarity is popular for floating point descriptors [Sidorov et al., 2014] and binary descriptors use the bit-wise Hamming measure [Hamming, 1950]. This process results in a set of data associations $\bar{\mathbf{x}}^j_{k-1} \leftrightarrow \bar{\mathbf{x}}^j_k$, where $\bar{\mathbf{x}}^j_{k-1}$ and

$\bar{\mathbf{x}}_k^j$ are the two keypoints which have been matched via their descriptors, $\mathbf{d}_{k-1}^j$ and $\mathbf{d}_k^j$. This comparison is also used for training vocabulary models, as we will discuss in Section 2.1.3.

A detailed discussion of the favourability and drawbacks of a host of local image feature extractors to visual navigation in variable environments is available in [Krajník et al., 2017], in which it is argued that ephemeral invariance categories (e.g. scale, viewpoint, etc.) of most extractors are less important than their robustness to naturally-occurring changes due to the passing of seasons or fluctuations in lighting.

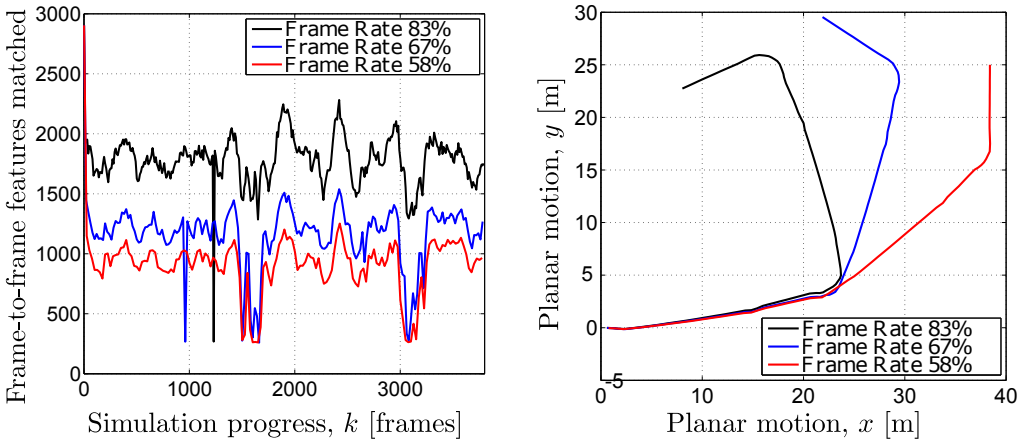
Robust model fitting RANdom SAmple and Consensus (RANSAC) is an iterative method which VO uses to propose a set of R of candidate transformations, $\mathbf{T}_{k-1,k}$, using a minimal number of randomly sampled prospective matches [Fischler and Bolles, 1981] where

$$R = \frac{\log(1-p)}{\log(1-e^m)} \tag{2.2}$$

These are ranked by the total number of associations, $\bar{\mathbf{x}}_{k-1}^j \leftrightarrow \bar{\mathbf{x}}_k^j,\ j = 1, 2, \ldots, m$, which satisfy the epipolar constraint to within a threshold, ψ, as introduced in Equation 4.8 such that prospective matches satisfying

$$\mathbf{e}_{k-1,k}^j = (\bar{\mathbf{x}}_{k-1}^j)^T \mathbf{F} \bar{\mathbf{x}}_k^j < \psi \tag{2.3}$$

are considered "inliers".



(a) The ebbing quantity of candidate features for matching disappear when consecutive floor images lose significant overlap.

(b) Local accuracy in mapping is retained as the homography estimation is starved of candidate features for matching, but drift worsens.

Figure 2.3: Degrading motion estimation accuracy when starved of features for matching and consequently inliers for refining pose estimates.

Figure 2.2 shows the inliers (green) and outliers (red) for a prospective set of matches (green and red) when used to estimate a constrained homography model as will be discussed in Section 2.1.2.

Example 2.1.1. *In Figures 2.3(a) and 2.3(b) we draw on Appendix A.1.1 to show matches discovered between subsequent frames and ground traces of a robot's motion respectively as it tracks the texture on the floor in front of its wheels. As the pose estimation is starved of inliers, achieved in this example by decimating the framerate such that there is an ever smaller overlap between frames of comparison, the quality of ego-motion recovered falters.*

Minimising reprojection errors While RANSAC rejects outliers faithfully, the output model is susceptible to noisy input data, and a non-linear least squares optimisation refines this estimate by minimising the re-projection

error of all data associations marked as inliers. As the camera observation model is non-linear [Hartley and Zisserman, 2003], this refinement is typically achieved by non-linear least squares estimation (LSE) techniques and gradient-based methods such as Gauss-Newton (GN) [Gauss, 1855] or Levenberg-Marquardt (LM) [Levenberg, 1944, Marquardt, 1963]. As these methods rely on quadratic costs, corrupted input data significantly degrades the quality of the resulting estimate, and M-estimation using the Huber function [Huber, 1964] is commonly employed to reduce the impact of large errors. In Figure 2.2 the longest diagonal red line shows such a prospective match which should be weighted and mitigated in this way.

2.1.2 Constrained Estimation

Oftentimes, as indicated in Example 2.1.1, the motion that a mounted camera undergoes is governed by a kinematic model which constrains and reduces the complexity of pose estimation. This is particularly true for field robots which are limited by the world, infrastructure, or the design of the actuators in robotic drive systems to moving in a restrained fashion – Appendix A.1 shows schematics for two robotic platforms with distinct kinematic behaviour.

Planar induced homographies Once again delaying mathematical detail to Section 4.1, as a consequence of the epipolar constraint, images of a point on the groundplane, $\bar{\mathbf{X}}^{\pi}$, are related by a three degree-of-freedom (3DoF) "planar-induced homography" such that $\bar{\mathbf{x}}_k = \bar{\mathbf{x}}_{k-1}$. It is shown in [Ma, 2004] that the relative pose between the subsequent frames is related to the ho-

mography as

$$\mathbf{H}_{k,k-1} = \mathbf{R}_{k-1,k} + \frac{1}{h}\mathbf{t}_{k-1,k}\mathbf{n}^{\pi T} \tag{2.4}$$

and that the singular-value decomposition (SVD) of the calibrated homography, $\mathbf{K}^T\mathbf{H}_{k,k-1}\mathbf{K} = V\Sigma V^T$, results in four possible solutions for the (up-to-scale, where the scale factor is controlled by a calibrated camera height, h) motion between the two frames and the plane normal, namely:

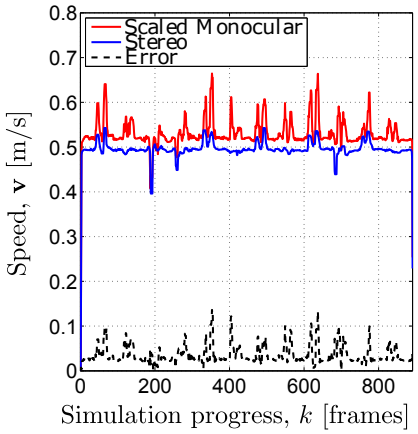
$$\{\mathbf{R}_{k-1,k},\ \frac{1}{h}\mathbf{t}_{k-1,k},\ \mathbf{n}^{\pi}\}_{i=1}^{i=4} \tag{2.5}$$

which are disambiguated by applying kinematic constraints which take into account the maximum angle that the normal can make with the plane as well as the maximum translation we expect frame-to-frame.

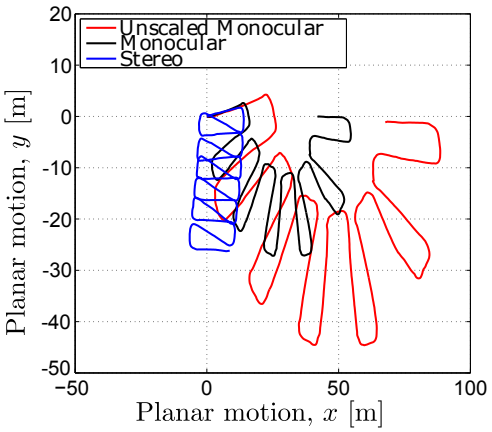
These visual servoing techniques are alone sufficient to enable localisation by tracking features over reasonably textured surfaces, as shown in [Blanc et al., 2005] with an upward-facing camera or with a downward-facing camera in [Gadd and Newman, 2015].

Example 2.1.2. *In Figure 2.4 we once again draw on the small-scale dataset of Appendix A.1.1 to provide a comparison of ego-motion estimation for a small robot which is limited to moving on a planar surface. We see that there is a consistent scale factor which distinguishes stereo techniques from monocular techniques.*

Motion planning and control, however, require a Euclidean characterisa-



(a) The average speed obtained by using a calibrated monocular camera behaves in a manner comparable with motion inference from stereo vision.



(b) Compared to stereo odometry, ground plane trajectories yielded by a monocular system are appear warped on a global scale, yet are locally accurate.

Figure 2.4: An example showing robustness and accuracy in motion inference and consequently maps on a local scale from monocular estimation strategies.

tion of the relationship between places. Moreover, as Examples 3.2.1 and 2.2.2 will show, while odometry estimation of this kind can be used within the EBN paradigm naturally, it is limited to working envelopes of only several hundred metres. Thus, while we may utilise these examples for ease of conveying ideas regarding localisation as a metric graph relationship – as in Section 4.2 to come – the stereo odometry estimation technique discussed in Section 2.1.1 above is inherent to the experiments performed in Chapters 4 to 6. Unless otherwise stated, the experiments presented in Chapters 4 to 6 make use of FAST corners and BRIEF descriptor patterns for VO. This is in contrast to the place recognition module to be introduced next and detailed further in Section 4.3.1, which utilises both the detection and description aspects of SURF.

2.1.3 Place Recognition

In contrast to the discussion thus far, topological techniques do not specify a metric (position and orientation) relationship between a robot's current location and a place in the world, but can be used to seed pose estimation.

Visual vocabularies The vector space model, used in text-based image retrieval, considers images as documents consisting of "visual words" [Sivic and Zisserman, 2003] such that each descriptor, $\mathbf{d}_j$, identified in all training image frames is associated with a "visual word", $\mathbf{w}_i$, in a "visual vocabulary"

$$\mathcal{V} = \{\mathbf{w}_i\}_{i=1}^{i=|\mathcal{V}|} \tag{2.6}$$

In general, the k-means [Duygulu et al., 2002] algorithm is used in constructing this vocabulary – clustering similar sparse features and weighting each word by its "inverse document frequency" (IDF) [Sparck Jones, 1972] given by

$$w_i = \log N - \log n_i \tag{2.7}$$

as a direct measure of their rates of occurrence, where N and n_i are the number of images used to construct the vocabulary and the number of occurrences of the i-th word in the training corpus, respectively. Here, words with a low IDF are more dispersed in the scene (e.g. common textures, such

as brickwork or shrubbery) and are thus less useful for place recognition.

Vocabulary encoding A visual image, $\mathcal{I}$, can be represented within this vocabulary model by a real-valued bag of words (BoW) vector which indicates the presence of visual words

$$\mathbf{b} = [u_1, \ldots, u_i, \ldots, u_{|\mathcal{V}|}]^T \tag{2.8}$$

Each BoW is populated by inspecting occurrences of every vocabulary element in an image and enforcing a minimum distance in descriptor space to the next-nearest vocabulary element [Ho and Newman, 2007]. The terms u_i for this BoW parameterisation are computed by weighting the word-specific IDF terms discussed above by the "term frequency" (TF-IDF) to account for the number of occurrences of the word in the image and the total number of words observed in the image [Sivic and Zisserman, 2003].

Scene similarity Using this encoding, a measure of similarity between the scenes represented by $\mathcal{I}_A$ and $\mathcal{I}_B$ can be computed as the pairwise cosine distance

$$\mathbf{V}(A, B) = \mathbf{b}_A \cdot \mathbf{b}_B \tag{2.9}$$

Computing similarity terms as is done in Equation 2.9 over a sequence of video images constitutes a Visual Similarity Matrix (VSM), $\mathbf{V}$, which

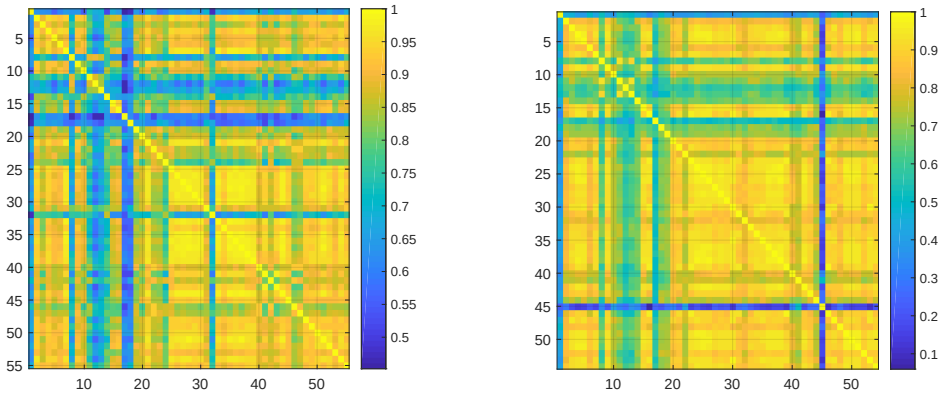


Figure 2.5: Pairwise, symmetric similarity matrix for two contiguous locations (ORI, left, and BEVINGTON, right) from Appendix A.2.2.

can be searched with dynamic programming techniques to find sequences for loop closing [Ho and Newman, 2007]. In this thesis, however, as shown in Figure 2.5, we use it as a measure of similarity between images from similar locations in the datasets presented in Appendix A.2.2, in order to distribute visually similar or dissimilar data to *fleet* members, such that the exchange and leveraging of valuable navigation data becomes clear when discussing the results presented in Section 6.4.9.

Example 2.1.3. *Consider the visual similarity matrices shown in Figure 2.5. Clearly, there are trends in the uniqueness of each robot's input to the DM, where logs eight to twelve are typical evening scenes, and as such are bonafide outliers. However, spurious appearance change must be taken into account in any approach to lifelong navigation, where log forty-five is an example of harsh illumination temporarily blinding that robot's viewpoint. Here each agent must remain flexible in managing and leveraging the trust they place in the expertise of their peers in and around the world.*

In contrast to the use of FAST corners and BRIEF descriptors in the VO implementation used throughout this thesis, the similarity terms shown in Figure 2.5 and used by the *canonical DM client* in Chapter 4 employ SURF features, which are highly discriminative in differentiating scene appearance [Han et al., 2017].

Generative models Using the cosine similarity of the form shown in Equation 2.9 above may become prohibitively expensive for large vocabularies. Indeed, the vocabulary model used in the experiments for Chapters 4 to 6 contains 2048 distinct words. Probabilistic Localization and Mapping in the Space of Appearance (FAB-MAP) [Cummins and Newman, 2008] approximates the high-dimensional discrete distribution of the co-occurence of visual words at places by learning a generative "chowliu tree" model [Chow and Liu, 1968] as an approximation of the joint probability in mutual observations of visual words to yield, at run-time, a likelihood that the location from which the query image is captured is being visited for the first time [Cummins and Newman, 2011]. FAB-MAP was shown over a 1000 km journey to scale well with the potentially vast scope of imagery as collected by a *fleet* of autonomous vehicles operating for indefinite periods on public roads. However, techniques such as these may still prove to be a bottleneck in performance due to the linearly increasing expense of a vocabulary lookup with map size. Deploying intelligently subsampled maps, as we will show in Chapter 6, alleviates this bottleneck.

2.2 Lifelong Navigation

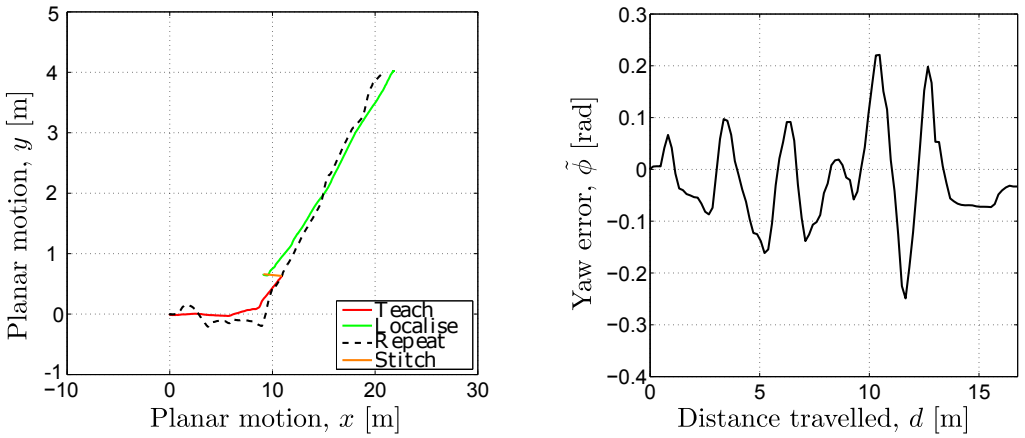
We discuss in this section a popular mapping and localisation framework which incorporates the motion estimation and localisation concepts discussed in Section 2.1 above and supports an operational framework for autonomous agents which we use within the experimental approach in Chapters 4 to 6.

2.2.1 Topometric Maps

Simultaneous Localisation and Mapping (SLAM) treats the environment as initially unknown while simultaneously localising within that environment [Dissanayake et al., 2001]. As the computational complexity of these techniques is dominated by the number of landmarks, processing performance degrades poorly with the extent of the environment and is limited in the most successful cases to small indoor environments [Klein and Murray, 2007] and outdoor environments with modest extent [Bailey and Durrant-Whyte, 2006]. Additionally, any inaccuracy in measurements causes the scale of the map to drift gradually change over time.

However, autonomous navigation requires metric accuracy over modest distances and only topological connectivity to farflung regions – suggesting that many meaningful robotic tasks require only "metric" localisation in the form of 6DoF pose with respect to a prior map [Sibley et al., 2010b]. Furthermore, this thesis is concerned with bolstering the urban driving capabilities of mobile robots over many repeated sorties along routes on the order of tens of kilometres.

Section 4.3.1 describes further how these maps are represented as graph



(a) (b)

Figure 2.6: Multiple passes (red and green) of overlapping regions of the same workspace, enables robots to automatically repeat (black, dashed) two segments taught at distinct times and linked (orange) as a byproduct of visual localisation.

structures. Generally, however, topometric maps are implemented as a collection of submaps, each of which models only the local environment, and are metrically joined such that motion planners know how to transition between them [Sibley et al., 2010a].

Example 2.2.1. *To motivate the use to topometric map representations, consider here a robot equipped with a downwards-facing camera is tracking the tarmac texture on the surface upon which it is driving, where more detail is given in Appendix A.1.1. Figure 2.6(a) shows how local submaps can be "stitched" together such that a robot can follow paths not necessarily in the order in which they were piloted.*

Importantly, these submaps are not optimised for global consistency and do not explicitly align in a unique co-ordinate frame, and as such map augmentation is done in constant time [Sibley et al., 2010b]. However, as dis-

cussed in Section 1.2 as a guiding question for this thesis, extended use of a regularly augmented map imposes an ever-increasing demand on CPU, disk, and network resources.

2.2.2 Visual Teach-and-Repeat

The manual creation and subsequent automatic reuse of paths through a robot's environment as introduced in Example 2.2.1 is known as teach-and-repeat (T&R) [Zhang and Kleeman, 2009] and provides a natural framework for learning appropriate behaviour in environments that robots may repeatedly traverse. Within an urban environment consistently visited by a *fleet* of autonomous vehicles, this paradigm is especially useful, as agents can treat any mapping information shared with them as they would the manual intervention inherent to T&R, which we show in Chapter 5 is potently beneficial to their localisation performance within regularly visited workspaces.

As an introduction, Figure 2.7 represents key aspects of the VT&R system presented in [Furgale and Barfoot, 2010] which demonstrated, by repeating 32 km of routes taught with a stereo camera with 99.6 % autonomy, the effectiveness of topometric maps.

Teach During a learning phase, the robot is piloted along a route within a section of its operational envelope, in this way being educated by operators during their normal tasks. The system encourages a comprehensive and interconnected representation of the world in the form of a large network grown over the course several such operator-led missions, and results in more accurate goal acquisition through an unrolling of localisation error

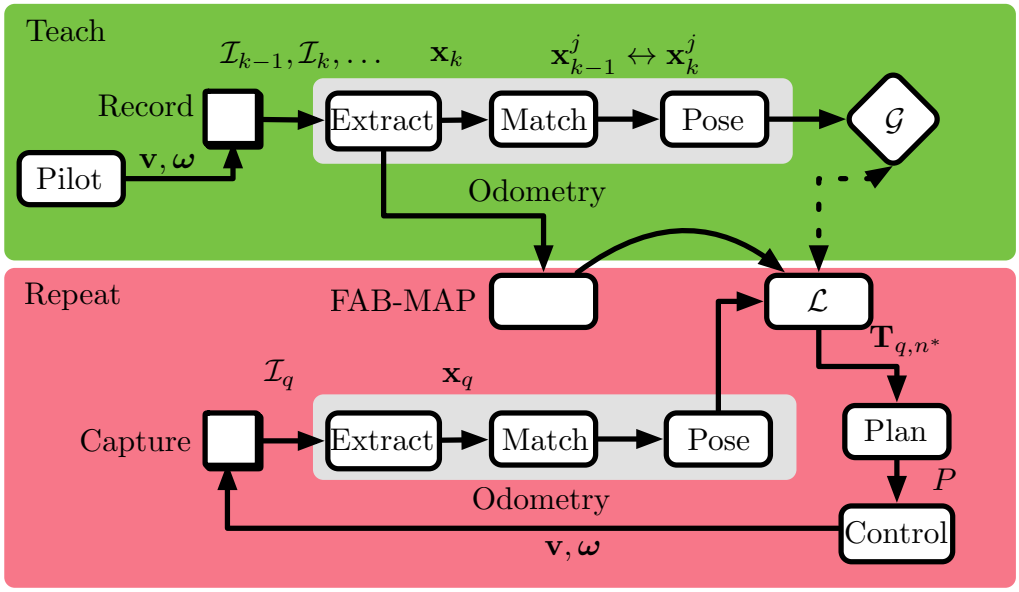


Figure 2.7: The VT&R framework is universal, with the central database granting read and write access to both mapping and localisation processes, allowing for large network representations of environments to be grown, over which space the robot can robustly localise and traverse autonomously (red). In this thesis, the sharing of mapping information can be understood as a substitute or proxy for the classical setup in which a human expert dictates the layout of mapping information by issuing motion commands (green).

when retreating along planned paths [Stenning et al., 2013].

Repeat During the repeat phase, the robot localises within this network of submaps using simple and efficient searches in the local neighbourhood around its current pose, a tool to be discussed in Section 4.1.2 and in use throughout this thesis. Planning in the network using this same search (which may be over a graph or tree, or an occupancy grid) allows the robot to autonomously navigate from one point of interest to another, not necessarily via a path that it was previously taught.

Other T&R systems do not require precise localisation within topomet-

ric maps and simply "replay" the taught velocities while steering in order to correct heading relative to the intended route [Krajnik et al., 2017]. An example of such yaw-based control is shown in Figure 2.6(b).

2.2.3 Experience-Based Navigation

Topometric mapping and localisation for T&R, particularly in the combinatorial space of appearance and on the spatial scale of data collected over the lifetime operation of *fleet*s of robots, demand the provision of some robustness to unmodelled modes of appearance change.

Summarisation approaches Aggregation of scene variation as "Summary Maps" merges different appearance conditions into a single, multi-session representation [Mühlfellner et al., 2016]. View deletion as a process for deprecation of keyrames from a skeleton map [Konolige and Bowman, 2009] bears some resemblance to the localiser-driven *commutation policy* to be discussed in Section 4.4 by means of the versioning notation in Section 3.1.

Composite approaches In contrast to these approaches, Experience-Based Navigation (EBN) curates a database of distinct visual experiences which in concert represent change in the world– avoiding ill-posed attempts at capturing an averaged version of everything ever seen, and side stepping the difficult issue of data fusion in the long-term [Churchill and Newman, 2012a, Churchill and Newman, 2013].

Example 2.2.2. *Figure 2.8 shows some example experience maps built by a robot tracking the perceived texture of various surfaces in front of its wheels,*

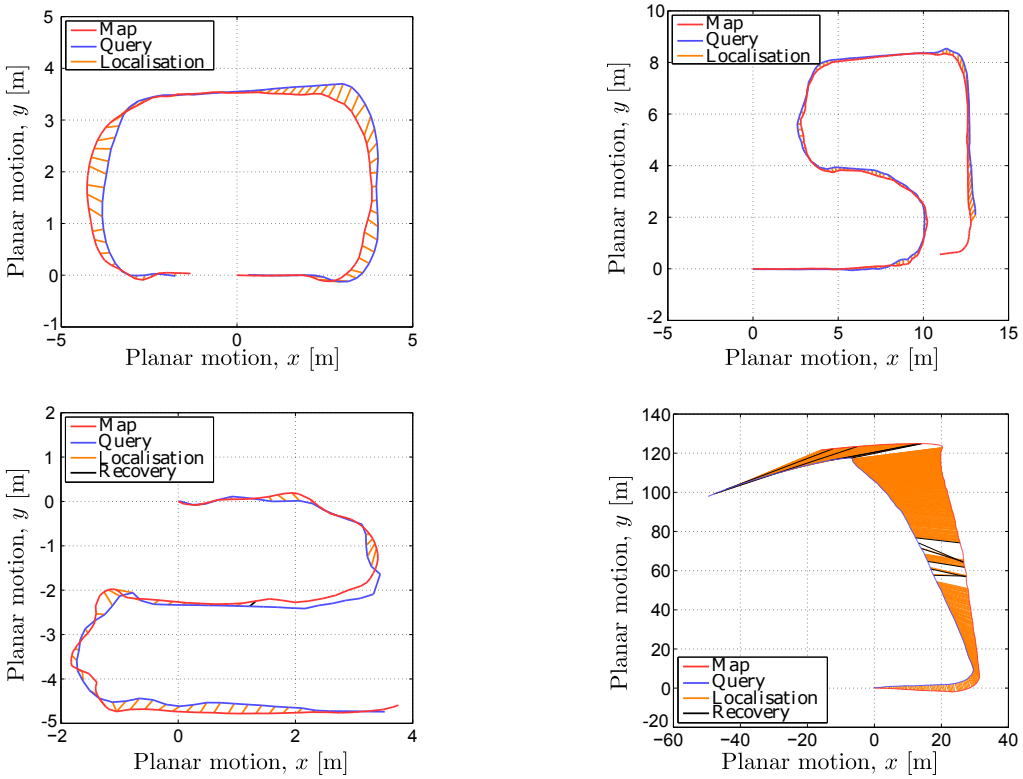


Figure 2.8: An implementation of monocular EBN is able to gracefully recover (black) and can resume normal localisation (orange) but localisation regularly fails in the face of changes in lighting or unmodelled vehicle movement, and these experience mappers are typically limited to workspace envelopes that are no more extensive than several hundred metres (e.g. 200 m, bottom right).

as detailed in Appendix A.1.1, and built as a composite representation by fusing into the map each experience as it is recorded.

Importantly, the experience mapper of [Linegar et al., 2015] is used in this thesis's implementation in Chapter 4 as it falls within the class of algorithms which not only gather overlapping representations of the world but instigate database writes *only* when the state of the map is not sufficient to explain the robot's current experience of a place (i.e. when localisation fails

sufficiently badly) [Churchill and Newman, 2013], in contrast to the EBN implementation of [Gadd and Newman, 2015] as discussed in Example 2.2.2.

Sequence-based approaches Recent work has involved searching for consistent sequences of image matches [Milford and Wyeth, 2012], which has been extended to matching places from day to night by the use of sky-darkening techniques [Pepperell et al., 2014]. Inferring average scene depth along traverses and rescaling views of the same place at different depths is shown in [Pepperell et al., 2015] to maximise their similarity. Formulation of image matching as a minimum cost flow problem in a data association graph is helpful in dealing with non-matching image sequences caused by transient occlusions [Naseer et al., 2014].

Section 2.2.4 discusses condition-invariant and condition-predictive image processing and mapping techniques further.

2.2.4 Predictive Approaches

Vision-based localisation is adversely affected by typically strong visual features which cause considerable change in the appearance of a place, such as shadows. Techniques which learn a model from a source data distribution which performs well on a related target data distribution can be used to mitigate this.

Condition-invariance [Corke et al., 2013] describes conversion of colour images of the scene to greyscale invariant images in which pixel intensities are a function of underlying material property instead of lighting. Distinct from

the visual vocabularies discussed in Section 2.1.3 above, [Neubert et al., 2013] present appearance change prediction (ACP) for learning how the visual appearance of an environment changes over time using a dictionary to translate between vocabularies of superpixels (SP-ACP).

Domain Adaptation Convolutional neural nets (CNNs), specifically Generative Adversarial Networks (GANs) automatically learn losses that adapt to the data and and can be used for a variety of image-to-image translation problems [Isola et al., 2017] to synthesise images of the same location under different conditions (e.g. "winter" to "summer"). Indeed, [Porav et al., 2018] fine-tune these types of image synthesisers using losses on differentiable outputs of the feature detector and descriptor pipeline inherent to metric localisation.

Spectral techniques [Rosen et al., 2016] use Bayesian-based persistency checks to predict which scene elements will be visible after some time and which are likely to disappear from view. In [Krajník et al., 2016] a FreMen (Frequency-Map Enhanced) spatio-temporal occupancy grid explicitly represents persistence and periodicity to predict the probability of future environment states as functions of time in order to curate mission-specific maps for localisation and planning.

Section 6.1.2 discusses a temporal filter for data exchanges based on a centrally assembled topometric map and co-localisation statistics over the course of fleet operations.

2.2.5 Collaborative Localisation

In contrast to the single-agent case presented in Section 2.2.3 above, multi-agent systems (MAS) sharing exteroceptive measurements can be exploited to add redundancy and improve the comprehensiveness of the world representation. We will show in Section 5.1 that a *central authority* can correlate experiences as individually collected by agents in the *fleet*. Later in Chapter 5, however, we depart from this centralised model to a totally distributed visual localisation scheme which will scale with the size of the *fleet* more gracefully.

Centralised approaches Multi-robot, keyframe-based SLAM is serviceable by estimation of the relative position of robots in a central frame of reference grown by the fusion of individual maps [Riazuelo et al., 2014]. Collaborative navigation based on the Extended Kalman Filter (EKF) with interleaved update algorithms [Bahr et al., 2009] and central optimal estimation of all robot poses [Bailey et al., 2011] both require robust and efficient communication of robot-to-robot pose information [Cunningham et al., 2012], interfaces for which we will discuss in Section 5.3. In [Forster et al., 2013] the problem is divorced from the estimation algorithm by the design of data structures and processes for thread-safe modification of the central map by local maps constructed with generalised ego-motion estimation and place recognition, but a central map assembly platform is still relied upon.

Distributed approaches Distributed estimation using the EKF are a means of escaping the *central authority* [Carrillo-Arce et al., 2013] but are

subject to a unimodal distribution of the state estimate as a result of the required linearisation, which is addressed by probabilistic modelling of pairwise interactions in [Fox et al., 2000] by the use of particle filters, which can universally approximate arbitrary probability distributions.

Ceilings in communication range and bandwidth are surpassed by marginalising agent-specific factors from batch-summarised local maps before sharing [Cunningham et al., 2013]. Summarisation, however, preempts individually held but globally consistent maps and [Cieslewski and Scaramuzza, 2017] propose a candidate selection for choosing a single robot to which a full query is sent based on a visual bag-of-words representation that can be distributed by pre-assigning words of the vocabulary to different robots.

Auctioning approaches A combinatorial space of interactions of this kind can be explored by role-assignment within a *fleet* behaving under common auction policies [Frias-Martinez et al., 2004]. Auctions require some measure of the quality of trades, which is facilitated in [Wang et al., 2016] by the use of a pricing model in a leader-follower regime. Chapter 6 suggests a pricing model for the data structures and interfaces developed in Chapters 4 and 5 which is tailored to meaningful pairwise exchanges of navigation data and presents the experiments in Section 6.4 as variations of simple first-price sealed-bid (FPSBA) or blind auctions in which all bidders simultaneously submit sealed bids, so that no bidder knows the bid of any other participant [McAfee and McMillan, 1987]. In the centralised case, this simplifies to dispensing portions of a privileged map which are most relevant to a mission.

2.2.6 Map Maintenance

This section provides background motivation for DM agents to limit the size of the map used for online localisation and mapping.

Update and refinement Memory-based approaches identify locations that should remain in a working – or short-term – memory (WM/STM) for online processing from locations that should be transferred to a Long-Term Memory (LTM) store [Labbé and Michaud, 2018]. In [Biber et al., 2005] a sample-based representation in which LTM and STM interact at different rates depending on the timescale, tracking both stationary and non-stationary (moving objects and structural changes) environmental elements.

Map sharing The robotics community relies inherently on Version Control Systems (VCS) to manage file history within software projects [Wingerd, 2005]. Indeed, approaches to collaborative localisation (CL) are increasingly taking the form of "torrent" tools for web-based distributed map management borrowing from the vocabulary of software versioning and management. Using Centralised VCS, such as the *central authority* investigated in Chapter 5, a master repository maintains a history of all incremental changes and consolidates the changes from subordinates [Nagel, 2005]. Decentralised systems, however, dictate that suboordinates own their own copies of the repository, where updates take place in the form of any number of interactions between every agent [Loeliger and McCullough, 2012].

VCS approaches In [Cieslewski et al., 2015], in a similar in spirit to the centralised approach to be discussed in Chapter 5, Optimistic Concurrency Control (OCC) forces processes that concurrently operate on some data to operate on their own copies before merging back to the reference state. This results in regular exchanges of the full map data between all robots, incurring large data transfers which [Cieslewski and Scaramuzza, 2017] handle by a data-efficient method for visual feature association with a minimum amount of data which is linear with trajectory overlap. Chapter 4 performs this same style of exploitation of perceived relevance among peers for versioning of navigation data, while the *product policy* introduced in Section 3.4 is used to demarcate the scope of these versioning operations to compact regions of interest in the world, where Figure 2.8 shows such an example demarcation for three connected areas of a warehouse environment (top left, top right, and bottom left).

2.3 Localisation Metrics

This section introduces the localisation metrics used to prove system capabilities in Chapters 4 to 6. Additionally, this section concludes the chapter by describing the data-driven validation schemes used throughout this thesis.

2.3.1 Accumulating Failures

Here, a reasonable concern with regards to localisation capability is the anticipated period over which a robot can be expected to travel without successful localisation, and becomes vulnerable to drift in VO or, in the worst case, slip

in wheel odometry. Localisation failure is detected automatically when the agent in question fails to report a pose registered to the underlying map.

Failure modes Here, these failures are a result of a drastic change in appearance, occlusion, or viewpoint and manifest within the localisation system by way of any one of the subsystems detailed further in Chapter 4, including:

1. The minimisation of reprojection errors, as discussed in Section 2.1.1, failing to converge, or

2. Convergence is achieved, but an inlier threshold is not satisfied.

Counting failures We require a metric for localisation robustness which summarises performance over vast-scales and highlights problematic areas where the system failed. Metrics such as precision-recall (PR) curves are commonly used in evaluating place recognition systems but essentially consider the percentage of frames not localised (PNL). These are as such an incomplete reflection on the performance of the autonomous agent as during drop-out the robot is able to integrate its position and orientation from the last reported pose and a likely corridor around a taught trajectory within which a vision-based localisation system is still able to localise exists [Dequaire et al., 2016]. Beyond simple PNL measures, therefore, this thesis considers the instances in which the robot fails to localise, $\mathbf{f}$, where each f_i represents a drop-out, measured in metres and computed by integrating the ego-motion as estimated by VO.

Accumulating failures In [McManus et al., 2014] this counting procedure is used to compute the probability of travelling a distance greater than f without satisfactory localisation as a ratio of the accumulated failures greater than f to the total distance travelled during testing

$$p(f_i \geq f) = \frac{1}{D} \sum_{f_j \geq f} f_j \tag{2.10}$$

where the lengths, $\mathbf{d}$, of well-localised portions, d_i, of the trajectory are used for computing the normalisation term

$$D = \sum_{f_j} f_j + \sum_{d_j} d_j \tag{2.11}$$

Once computed, this cumulative distribution can be further summarised by an area-under-curve (AUC) statistic which indicates the average localisation failure length weighted by the empirically measured probabilities of failure of Equation 2.10. A good cumulative failure distribution will rapidly dwindle to zero from a moderate peak.

Example 2.3.1. *Consider Figure 2.9, taken from the small-scale samples in Appendix A.1, in which a robot equipped with an experience mapping framework is localising over maps built with various ground textures. Generally, the judgement of one distribution as better than another is dependent on the specific reasons for failure, but is readily apparent in the ordering of the curves on a visual plot. Here, important aspects include the leading peak (the probability of failing even once), the lagging tail (the largest localisation failure), and the AUC as discussed above. In this example, performance over*

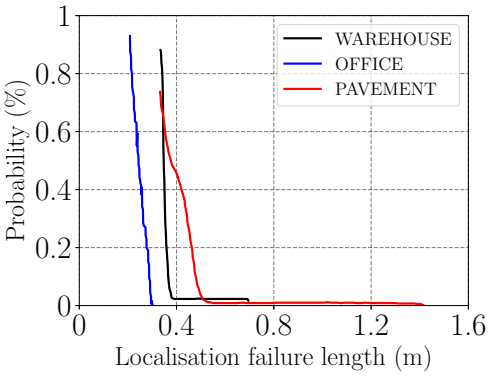


Figure 2.9: Cumulative localisation failure distribution showing the probability of travelling blind for more than a certain distance before resuming normal localisation behaviour, from [Gadd and Newman, 2015]. Here, localisation within the "Warehouse" map (black) fails maximally with a 0.70 m drop-out, and there is a 2.29 % probability of having to rely on dead-reckoning for distances in excess of 0.5 m.

the "pavement" texture seems superior by inspecting the peak, but exhibits a longer tail and as such a greater AUC.

A distinction must be made between counting and accumulating failures and evaluating failures on a held-out set which is common to all robots and is not added to the map under testing conditions, which we discuss next.

2.3.2 N-Fold Cross-Validation

In addressing these resource management concerns, we present experimental validation for arbitrarily-sized teams of agents (typically between two and five agents) which are each responsible for entering some portion of the datasets presented in Appendix A into the DM. The behaviour of EBN itself, as well as that of the MAS extensions we detail in the following chapters, is highly dependent on the order of appearance-varying data that is injected into the

system.

Definition 2.3.1. *Any of the four datasets presented in Appendix A can be considered a set of M logs, $\mathcal{D} = \{\ell_1, \ldots, \ell_M\}$, where each log consists of the sensory information that agents require to populate their maps.*

To properly verify the utility of the sets of policies we present in the face of *a priori* unknown modes of appearance change, we present statistically robust experimental results by performing N-fold cross-validation, involving:

1. Dividing the totality of M logs into N groups of logs, $\mathcal{D} = \{\mathcal{D}_i\}_{i=1}^{N}$, and

2. Mapping and trading with the first $(N-1)$ of the groups, $\{\mathcal{D}_1, \ldots, \mathcal{D}_{N-1}\}$, which constitute the training sets[1], and

3. Localising in this map using the Nth group, $\mathcal{D}_N$, which is held-out as the test set.

We repeat this procedure N times (such that each group of logs is used as the held-out set exactly once), aggregating[2] all statistics (e.g. a cumulative distribution of localisation failures as discussed in Section 2.3.1) over the N iterations. This approach ensures that there is no biased selection of logs with related appearance that should localise well together.

[1]Either via single-agent EBN (sequentially, in their natural order) or various instantiations the policies presented throughout this thesis, where the $N-1$ groups represent the same number of robots which are responsible for the mapping of $\frac{|\mathcal{D}|}{N-1}$ logs.

[2]In the randomised scenarios, sample standard deviations are used to indicate statistical effects around a sampled mean, whereas the cross-validated scenarios present the mean over all of the N groups.

2.3.3 Randomised Trials

Alternatively, we can avoid bias in the arbitrary assignment of logs to each robot by aggregating randomised statistics which are accumulated over T *Monte Carlo* trials [Robert, 2004]. This approach, like that discussed in Section 2.3.2 above, should protect against bias of arbitrarily assigning logs with similar appearance, except when we explicitly state that we do not do so, as will become useful in assigning leading value providers over a *product* space in Chapter 6. If the number of robots in the team is denoted as $|\mathfrak{R}|$, each trial involves:

1. Dividing M logs into $|\mathfrak{R}|$ groups of logs, $\mathcal{D} = \{\mathcal{D}_i\}_{i=1}^{|\mathfrak{R}|}$, and
2. Where each of the $|\mathfrak{R}|$ groups is assigned (randomly) to a robot which becomes responsible for the mapping of $\frac{|\mathfrak{R}|}{|\mathcal{D}|}$ logs, and
3. Shuffling the order of each group, $\mathcal{D}_i$, and finally
4. Proceeding with mapping and/or trading[3] activities as discussed in Section 2.3.2.

Here, however, we are not using a held-out set, and the cumulative failure distributions discussed in Section 2.3.1 are a result of counting failures over the lifetime of operation of this simulated *fleet* and must be understood as such.

[3]where each robot in the experiment incrementally maps chunks of each dataset log, ℓ, within its dataset log group, $\mathcal{D}_i$, and pauses occasionally to make and respond to queries from other agents in the DM. A perspective of this as a "Multi-robot Finite-state Machine" is available in Chapter 5

2.4 Implications on Fleet Operations

While EBN approaches provide significant robustness to appearance change, they ultimately prove computationally demanding, as environments which are variegated in their appearance may require many overlapping experiences to capture the full spectrum of change. Additionally, collaboration as discussed in Section 2.2.5 expands the workflow in Figure 1.2 with communication capabilities, indicated by the red, green, and blue shaded areas. This section thus discusses the counterproductive impact upon fleet operations – particularly disk, CPU, and network – of the gainful sharing of localisation capability, to motivate the VCS interpretation that follows in Chapter 3.

Disk The consequential explosion in experience density required to capture complex spectra of change (and the associated processing load) are mitigated by strategies including summarised (reduced) map representations [Dymczyk et al., 2015], prioritising the utility of past experiences to predict the real-time relevance of memories [Linegar et al., 2015], and induced forgetfulness [Milford and Wyeth, 2010].

Example 2.4.1. *Consider the experimental evidence from [Churchill, 2012]. The EBN system saves* 24.80 % *of all the possible experiences and is lost* 24.80 % *of the time when a single experience is used to represent the environment. Increasing the number of experiences improves localisation to a drop-out of* 14.12 %*, but the disk usage worsens to* 31.95 %*.*

This example illustrates the tension between considerations of disk management and desired localisation performance when designing EBN systems.

Expanding the workflow of experience mappers as depicted in Figure 1.2 will in effect increase the number of localisers present in the EBN system, and we present strategies for managing this trade-off in Chapter 5, experimentally verified in Section 5.4.

Example 2.4.2. *Considering experimental evidence from [Churchill, 2012] again, the EBN system improves its localisation drop-out from* 23.28 % *to* 14.12 % *when enriching the map by adding loop closures between experiences. Simultaneously, disk usage improves from* 50.53 % *to* 31.95 %.

In Chapter 6 we present a simple trading application of the combined mapping, versioning, and distribution system which is experimentally validated in Section 6.4. Here, as indicated in Example 2.4.2, it is urgent to ensure that all possible pairwise matches between experiences are added to the map, thereby reducing disk load and bolstering localisation performance.

CPU As experience density increases, the machine which a robot is equipped with must (on average) do more work to obtain a successful localisation. Overcoming experience density by attempting matches in a small set of experiences commonly localised against together as in [Linegar et al., 2015] may still result in longer average times to localise particularly when regaining localisation from a lost state.

Example 2.4.3. *Experimental evidence from [Linegar, 2016] shows that as the map used by the EBN system increases in size from* 2000 *to* 3000 *nodes, the probability of travelling more than* 10 m *without localisation degrades from* 14.4 % *to* 16.78 % *under a distance-based experience-selection mechanism.*

This is mitigated by an informed ranking policy which learns how to best use experiences as the map grows – actually improving localisation performance from 15.7 % *to* 3.6 %.

Generally, Example 2.4.3 motivates the *throttling* and *ranking* techniques we employ for subsampling of mission-relevant content before distribution through the DM in Chapter 6. Specifically, when communication amongst the *fleet* is arranged in the centralised fashion to be discussed in Section 5.1 we employ the experience-selection mechanism of [Linegar et al., 2015] and exploit its measure of experience co-occurence when distributing maps from a *central authority* which receives entire experiences from contributing agents.

Network Communication costs increase with the size of the *fleet* as well as with the size of environment and can be mitigated by optimal search policies based on appearance [Giamou et al., 2017].

Example 2.4.4. *[Cieslewski and Scaramuzza, 2017] provides experimental evidence that the data per query to a bag of words model increases linearly from* 100 kB *to* 700 kB *as the number of robots expands from one to seven. Pre-assigning vocabulary words to different robots ensures that the communication cost remains flat at* 100 kB.

Chapter 5 experimentally verifies a set of flexible interfaces which allow the *team* to engage in *ad hoc* trading configurations and mitigate this bandwidth effect.

Example 2.4.5. *[Giamou et al., 2017] show experimentally that as the candidate radius (for discovering loop closures) increases from* 5 m *to* 30 m *in*

a team of dialoguing agents, the communication cost worsens from 33 MB *to* 36 MB. *This is mitigated to* 31.6 MB *or* 16 MB *by including lightweight field of view (FOV) overlap and BoW measures of appearance similarity in planning for communication, respectively.*

The outbound (communicative) workflow aspects shown in Figure 1.2 are assumed in this thesis to be consistently affirmative (no network dropout is accounted for), except in the case that an agent decides that localisation performance for the section of world that it is interested in is sufficiently good. If we consider that the urban dataset detailed in Appendix A.2.2 is collected over a physical area which is approximately eighteen to twenty times more extensive than that of Example 2.4.5, the communication cost is expected to be in the hundreds of megabytes for the transfer of just a single experience over the entire route, which is not possible in realtime for field robots using WiFi connections. To this end, Chapter 3 introduces the *product policy* as a means of itemising data exchanges, which limits the scope of versioning operations on experience maps in Chapter 4 and is used more extensively in the trading application of Chapter 6.

2.5 Summary

This chapter provided prerequisite knowledge for the large-scale, lifelong, vision-only localisation problem. Basic camera properties were introduced in Section 4.1. Here, the properties of rigid-body transformations and the homogeneous co-ordinate systems they act upon were described.

Section 2.1 presented VO as a technique for incrementally determining

the ego-motion of a robot – where feature detectors and feature descriptors, outlier rejection, and pose estimation from point correspondences were illustrated in Examples 2.1.1 and 2.1.2 with simple visual servoing techniques. Topological localisation with place recognition was introduced and used to motivate the discussion around localising under conditions of extreme appearance change.

Section 2.2.1 introduced topometric mapping representations which integrate ego-motion estimation and loop-closing systems for lifelong navigation. Section 2.2.2 illustrated how these representations can be applied to robot autonomy through the VT&R framework and was followed in Section 2.2.3 by an introduction to the EBN paradigm as a counterpart to this autonomous framework. Section 2.2.5 initiated the discussion on the relationship between agents' experiences of the world and reviewed existing centralised, distributed, and VCS approaches to the multi-agent localisation problem.

Concluding this chapter and motivating the experimental outcomes for this thesis, Sections 2.3 and 2.4 discussed the impacts of the EBN paradigm on the operational reliability of teams of collaborating agents in terms localisation, disk usage, CPU load, and network traffic.

Chapter 3

An Introduction to Version Control and Patch Theory

Visual navigation data, representative of the lifetime of experience of a single agent, should not be considered in isolation from similar and potentially useful knowledge held by the *fleet* in concert. This chapter therefore puts forward representational requirements which are malleable to both of the centralised and distributed trading configurations of Chapter 5 and which mapping clients using the DM must conform to in order to ensure the changes they make to their maps can be effectively communicated with one another. To this end, while representations for robust single-agent visual localisation abound in the literature, this chapter provides a VCS encoding of common data structures employed in large-scale, lifelong navigation before the instantiation of a state-of-the-art experience mapper through this notation in Chapter 4 and proof of adherence to the requirements made here. Hence, this chapter introduces the formal theories of patches in terms of basic revision

control mechanisms and their use in evaluating VCS correctness. Alternative representations make use of cloud-enabled graph structures for separation of computation and data [Fourie et al., 2017] traversed and modified by a variety of different agents. In contrast, this chapter describes manipulations to local spatio-temporal indexed navigation data as hosted by each subordinate agent, and introduces the lightweight *product policy* as the only universally agreed upon shared data persistence layer common to all DM agents and as a means for itemising data exchanges for the trading application of Chapter 6.

3.1 Basic Elements of Patch Theory

Before dictating requirements for the underlying data that will allow distributed DM agents to compare and consolidate experience whenever they interact this section introduces versioning activities intuitively as syntactic representations of functions that change a repository [Angiuli et al., 2014]. We begin by drawing on the notation of [Dagit, 2009] to explain some basic elements of the *Patch Theory* used in evaluating VCS correctness within the Darcs Advanced Revision Control System (Darcs) community[1]. Specifically, Darcs considers patches as first-class citizens, where a repository can be viewed as a set of patches which form a partially ordered set (poset) [Dushnik and Miller, 1941], and is implemented in the functional programming language "Haskell" [Thompson, 1999][2]. Specifically, these changes are stored internally in a rigid order. However, a fundamental requirement in Darcs is the rearrangement of updates. Along with the notion of undoing

[1] `www.darcs.net/Theory`
[2] `www.haskell.org`

edits, reordering is the basis for implementing the merge tool that Darcs provides to its users. The repository and branching models of Sections 3.2 and 3.3 refer to this abstract combination of reverting and reordering as the *commutation policy*, before it is given an interpretation through the EBN framework in Chapter 4.

Repository The *repository representation* will be discussed in more detail in Section 3.2, describing the working copy and changes to it as entries into a database schema which lends itself well to lifelong mapping. Section 3.3 discusses how generalised changes to these database records as a result of pairwise trades between DM agents are tracked alongside the database itself. For now, however, consider a generalised repository, $\mathcal{R}_i$, which tracks a sequence, $\mathcal{H}_i$, of incremental changes to a working copy, $\mathcal{G}_i$, of some data as held by an agent, r_i, the current state or context of which can be uniquely identified as $\overset{a}{\Xi}$.

Patch updates A patch, ${}^{\overset{a}{\Xi}}\mathbf{P}^{\overset{b}{\Xi}}$, is a concrete representation of a transformation made to the state of the repository, identified by input and output states, $\overset{a}{\Xi}$ and $\overset{b}{\Xi}$. Here $\overset{a}{\Xi}$ is the context that the patch originates from and $\overset{b}{\Xi}$ is the context resulting from the patch.

Patch equality Two patches, ${}^{\overset{a}{\Xi}}\mathbf{P}^{\overset{b}{\Xi}}$ and ${}^{\overset{c}{\Xi}}\mathbf{P}^{\overset{d}{\Xi}}$, describe the same transformation of state if $\overset{d}{\Xi}$ is equivalent to $\overset{b}{\Xi}$ whenever $\overset{c}{\Xi}$ is equivalent to $\overset{a}{\Xi}$.

Patch composition Patches with coincidental output and input states, ${}^{\overset{a}{\Xi}}\mathbf{P}^{\overset{b}{\Xi}}$ and ${}^{\overset{b}{\Xi}}\mathbf{P}^{\overset{c}{\Xi}}$, can be combined in sequence, and written as ${}^{\overset{a}{\Xi}}\mathbf{P}^{\overset{b}{\Xi}}\mathbf{P}^{\overset{c}{\Xi}} = {}^{\overset{a}{\Xi}}\mathbf{P}^{\overset{c}{\Xi}}$.

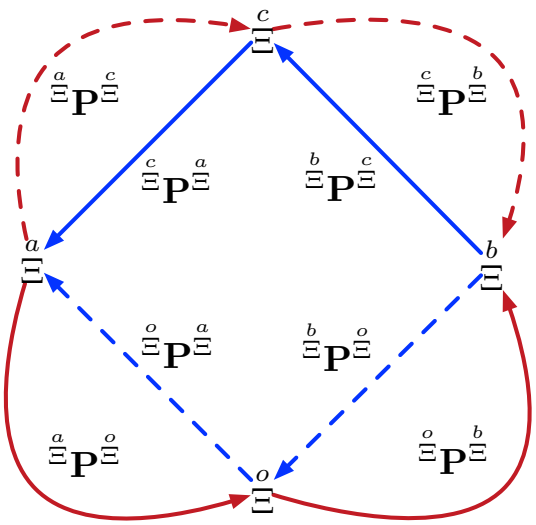


Figure 3.1: In general, a collection of unordered divergent *patch representations* can be consistently consolidated if the input and output *state representations* are monitored carefully and by the design of a *commutation policy*, Γ, which must take into account the underlying nature of the data which is being versioned.

Patch inverse If ${}^{\overset{a}{\Xi}}\mathbf{P}^{\overset{b}{\Xi}}$ is a patch, and ${}^{\overset{c}{\Xi}}\mathbf{P}^{\overset{d}{\Xi}}$ is another patch with the opposite transformation of state, we require that for every equivalence made between states $\overset{d}{\Xi}$ and $\overset{a}{\Xi}$, states $\overset{c}{\Xi}$ and $\overset{b}{\Xi}$ are also equivalent.

Patch commutativity Consider a pair of divergent (with equivalent input state, $\overset{o}{\Xi}$) patches ${}^{\overset{o}{\Xi}}\mathbf{P}^{\overset{a}{\Xi}}$ and ${}^{\overset{o}{\Xi}}\mathbf{P}^{\overset{b}{\Xi}}$, and some candidate patches, ${}^{\overset{a}{\Xi}}\mathbf{P}^{\overset{c}{\Xi}}$ and ${}^{\overset{b}{\Xi}}\mathbf{P}^{\overset{c}{\Xi}}$, the action of which should be convergent (towards an equivalent output state, $\overset{c}{\Xi}$), that is to say

$$ {}^{\overset{o}{\Xi}}\mathbf{P}^{\overset{a}{\Xi}}\mathbf{P}^{\overset{c}{\Xi}} = {}^{\overset{o}{\Xi}}\mathbf{P}^{\overset{b}{\Xi}}\mathbf{P}^{\overset{c}{\Xi}} \tag{3.1} $$

These commutative pairs are illustrated in Figure 3.1 and the guarantee of a successful merge given their existence can be understood visually by

following the effects of patches and their inverses between common states $\overset{o}{\Xi}$ and $\overset{c}{\Xi}$. Here, solid paths indicate "patches being commuted", and dashed paths indicate "patches commuted to".

In terms of recorded database content in the form of experiences, we require the details of this merge to be able to replicate the behaviour of the interlocked EBN localisation and mapping modules. Specifically, the implementation of this merge through the *commutation policy* is wed to the EBN system in Section 4.4.2 so as to be able to delete pre-recorded views between exchanged patch updates. This ensures that experience data communicated by a peer is not wholly copied into the requesting agent's map – parts of it are disregarded as would have been the case if the requesting agent had processed the data through EBN itself.

Policy 3.1.1. *If updates to the graph representation can be encoded as a patch representation, this patch representation is invertible (the reverse effect of the patch representation can be characterised with high fidelity), and there is some symmetric "commutation policy",* Γ*, such that*

$$(\overset{a}{\Xi}\mathbf{P}^{\overset{c}{\Xi}},\ \overset{c}{\Xi}\mathbf{P}^{\overset{b}{\Xi}}) = \Gamma(\overset{a}{\Xi}\mathbf{P}^{\overset{o}{\Xi}},\ \overset{o}{\Xi}\mathbf{P}^{\overset{b}{\Xi}}) \tag{3.2}$$

(or "dashed red paths commute to solid red paths" as a verbal interpretation of the same in Figure 3.1) and equivalently

$$(\overset{b}{\Xi}\mathbf{P}^{\overset{c}{\Xi}},\ \overset{c}{\Xi}\mathbf{P}^{\overset{a}{\Xi}}) = \Gamma(\overset{b}{\Xi}\mathbf{P}^{\overset{o}{\Xi}},\ \overset{o}{\Xi}\mathbf{P}^{\overset{a}{\Xi}}) \tag{3.3}$$

(c.f. "solid blue paths commute to dashed blue paths", Figure 3.1), then

we can always merge divergent patches.

Chapter 4 will explore these requirements further using a concrete instantiation of a state-of-the-art experience mapper, which will allow us to more easily illustrate the inversion of patches (which must maintain integrity of the *graph representation*) as well as the relationship of the *commutation policy* to the localisation problem in general.

3.2 Repository Model

This section describes in more detail the working copy component of the repository model mentioned in Section 3.1 in terms of its database structure. Section 3.2.2 will discuss the formulation of the general patch effects denoted in Section 3.1 for databases of nodes and edges as well as the repository context which identifies the input and output conditions of the database.

Representation 3.2.1. *The history of the repository is encoded in a sequence of patches*

$$\overset{*}{\Xi}\mathcal{H}_i^{\overset{d}{\Xi}} = \{\overset{*}{\Xi}\mathbf{P}^{\overset{a}{\Xi}},\ \overset{a}{\Xi}\mathbf{P}^{\overset{b}{\Xi}},\ \ldots\ ,\ \overset{c}{\Xi}\mathbf{P}^{\overset{d}{\Xi}}\} \tag{3.4}$$

which is referred to as the history representation.

Here, as will be discussed in Section 3.3.2, the *state representation* $\overset{*}{\Xi}$ indicates a pristine working copy with an empty map. Section 3.2.1 describes the representation for the working copy of the database, which (after performing a mapping run) will include database content not yet registered to the *history representation.*

Representation 3.2.2. *Agents in the DM navigate through the world by means of a directed "graph", "database", or "map" (terms which we will use interchangeably):*

$$\mathcal{G} = \{V,\ E\} \tag{3.5}$$

consisting of a set of vertices or nodes, $V = \{v_i\}_{i=1}^{|V|}$*, and arcs or edges,* $E = \{{}^{i}e_j = e_k\}_{k=1}^{|E|}$*, the combination to which we refer as the graph representation.*

For now, these two representations are combined to completely specify the condition of and history of changes to each agent's store of processed navigation data.

Representation 3.2.3. *Each agent,* r_i*, in the fleet possesses a repository*

$$\mathcal{R}_i^{\overset{a}{\Xi}} = \{\mathcal{G}_i^{\overset{a}{\Xi}},\ {}^{\overset{o}{\Xi}}\mathcal{H}_i^{\overset{a}{\Xi}}\} \tag{3.6}$$

which we refer to as the repository representation.

Section 3.3 completes the *repository representation* by discussing the *history representation* as a composition of changes to database records arranged in sequence.

3.2.1 Working Copy

An example of a generalised topometric map as a network of interconnected places (c.f. Section 2.2.1) is shown in Figure 3.2. This *graph representation* for maps lends itself well to both linear chains of ego-motion (c.f. Section 2.1.1)

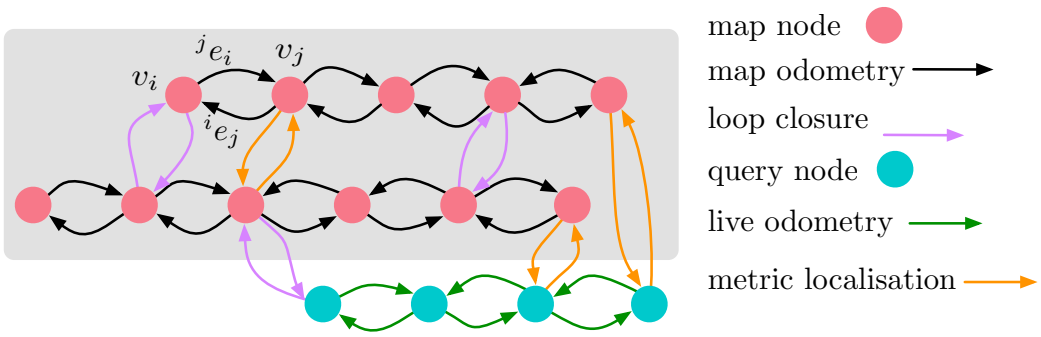


Figure 3.2: Nodes store a representation of distinct places, which may be a visual image, laser scan, etc. Edges indicate the connectedness of the world locally and store an appropriate relationship between places, which may be a fundamental or essential matrix, metric pose, wheel odometry, etc. These schema identifiers are accompanied by representations (Υ and ξ) which associate the database entries with the sensor records saved to disk separately from the database itself. Edge and node types are shown here for visualisation purposes only (e.g. metric localisation edges discovered as an agent records a video stream, or *post hoc* loop closure edges discovered offline) and are not considered distinct by the DM paradigm.

and topologically connected maps that have been amended with loop closures (c.f. Section 2.1.3).

In a manner compatible with all graph-based SLAM techniques, each element in the set of map nodes, V, must store appropriate representations of distinct places in the world, which correspond to measurements acquired at those positions [Grisetti et al., 2010]. Edges, however, impose spatial constraints between places, and result from observations or odometry measurements. The foundational concepts supporting the DM are ubiquitous to these representations, so long as agents interacting with one another have a common understanding of the way that these maps are curated.

Representation 3.2.4. *Content in the DM shall take the form of database nodes, v_i, which can be queried for all data relevant to the navigation problem*

as

$$\Upsilon_i = \mathcal{G}(v_i) \tag{3.7}$$

which we call the node representation.

Here, the exact composition of the *node representation* in terms of the sensor data attributed to it is detailed for an EBN implementation in Chapter 4. In the same general sense, directed edges between nodes store an invertible relational operator between places.

Representation 3.2.5. *All data relevant to a relation between DM content as referenced to nodes v_i and v_j as held individually by one or more agents in their graph representations can be queried as*

$$^i\xi_j = \xi_k = \mathcal{G}(^ie_j) = \mathcal{G}(e_k) \tag{3.8}$$

and is called the edge representation.

The skeleton database structure shown in Figure 3.2 depicts the working copy of the database using node and edge identifiers (v and e respectively) rather than the *node representation* or *edge representation* (Υ and ξ respectively). Indeed, the following sections formulate the format of the navigation data to be versioned in terms of the basic elements of patch theory discussed above in a similar fashion. The framework has ubiquitous applicability to systems localising in *graph representations* inspired by somewhat different sensor modalities, such as the LIDAR-based work of [Maddern et al., 2015],

but is validated in Chapter 4 by a clear improvement on single-agent EBN using stereo cameras.

3.2.2 Repository Context

This section defines the context of the repository indicated in Equation 3.6 (marking the map as being in a state, $\overset{a}{\Xi}$, i.e. $\mathcal{G}^{\overset{a}{\Xi}}$) by the identifiers on the *node representation* and *edge representation* which are stored in the working copy of a database.

Representation 3.2.6. *DM agents summarise all incoming and outgoing map content as a set of nodes and the edges which relate them as*

$$\Xi = \{p_i \mid p_i = (\iota_i,\ g_i)\}_{i=1}^{|\Xi|} \tag{3.9}$$

and which we refer to as the state representation.

Here, each *patch element*, p_i, groups a node (or vertex) and its adjacency list (incoming and outgoing edges) as a *graph element*

$$g_i = (v_i,\ \text{æ}_i) \tag{3.10}$$

where we have indicated the adjacency list, æ_i, as all incoming and outgoing edges with respect to v_i. A complete formalisation of versioning operations as they apply to topometric maps requires the facility to remove or deprecate database content and as such we describe the *patch element*

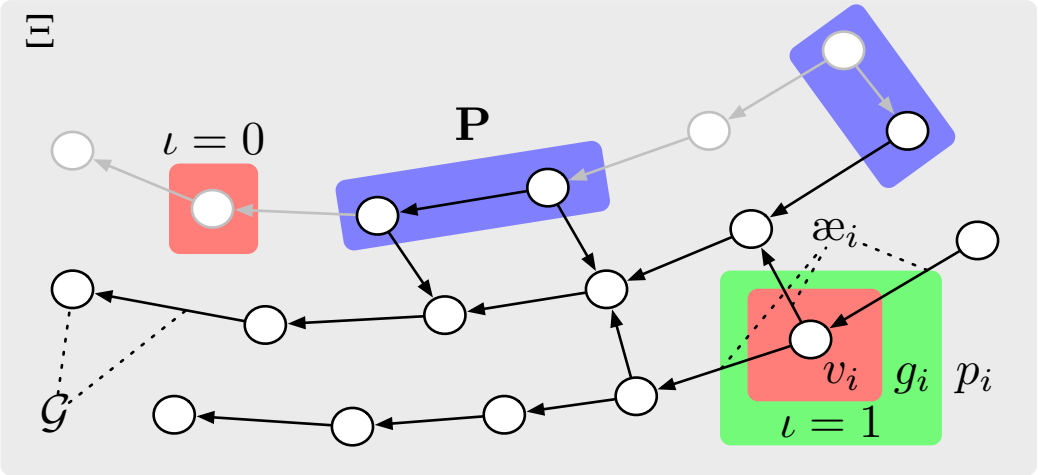


Figure 3.3: The repository state (grey shaded area) captures all additions to or removals from the database schema, and may include map information which is no longer present in the *graph representation*. In either case, atomic *patch elements* (green) anchor these changes to their node of reference, as well as the edges which originate from or terminate at that node. Patches (blue) update both graph and state representations and may consist of inserted patch elements (solid black nodes and edges) as well as deleted patch elements (faded grey nodes and edges).

action of each *patch element* upon the *graph representation* as

$$\iota_i = \begin{cases} 1, & \text{if } v_i \in \mathcal{G} \cap e_k \in \mathcal{G} \; \forall \; e_k \in \text{æ}_i \\ 0, & \text{otherwise} \end{cases} \tag{3.11}$$

which correspond to an insertion or removal of database content respectively. Importantly, the implementation of this *state representation* requires a lightweight summarisation of content and connectivity using only the identifiers, v and e, and not the complete set of sensor data hosted by the *node representation* or *edge representation*. Figure 3.3 shows the state of a working copy (grey, shaded area) that results from both additive and destructive queries to the database.

3.2.3 Encoding Map Updates

Map updates which result from a single DM agent processing a dataset log as discussed in Sections 2.3.2 and 2.3.3 (c.f. typical workflow grey shaded area, Figure 1.2), take the form of patches which are purely additive to the *state representation*.

Representation 3.2.7. *When a robot, r_i, executes a foray into the world, the novel content it records is represented as a patch, ${}^{\overset{a}{\Xi}}\mathbf{P}_i^{\overset{b}{\Xi}}$, which we call the patch representation, consisting of a set of atomic patch elements such that the map changes from state $\overset{a}{\Xi}$ to state $\overset{b}{\Xi}$ as*

$$\mathcal{G}_i^{\overset{b}{\Xi}} = \mathcal{G}_i^{\overset{a}{\Xi}} \cup {}^{\overset{a}{\Xi}}\mathbf{P}_i^{\overset{b}{\Xi}},\ \iota = 1\ \forall\ p = (\iota,\ g) \in {}^{\overset{a}{\Xi}}\mathbf{P}_i^{\overset{b}{\Xi}} \tag{3.12}$$

Alongside the working copy update, we maintain this increment to insular database content through the *state representation* as

$$\overset{b}{\Xi} = \overset{a}{\Xi} \cup {}^{\overset{a}{\Xi}}\mathbf{P}_i^{\overset{b}{\Xi}} \tag{3.13}$$

Over the lifetime of its operation, a single agent or EBN process updates the *graph representation* and *state representation* during many visits to the world (c.f. Section 2.2).

Example 3.2.1. *We show in Figure 3.4 a set of ground traces which correspond to a graph representation in use by a small robot which is accumulating experiences of the texture in front of its wheels over a* 24 h *period. Here, each of the coloured lines may be considered as a patch update, each of which contains patch elements which include edges to nodes which have been included*

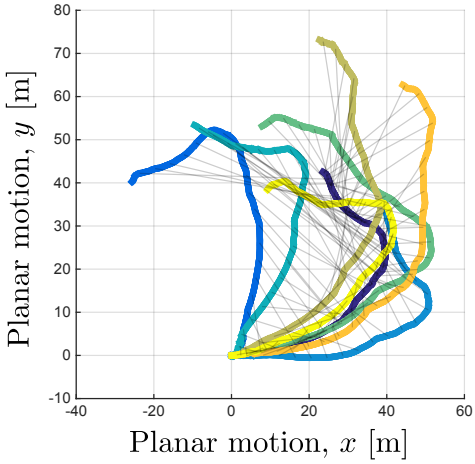


Figure 3.4: Example experience map built using planar-induced homographies [Yeomans et al., 2017]. The grey lines connecting these colourful traces indicate the localisation of one foray (experience) to another. Each experience, or portions of experiences, are considered as patches effected upon the *graph representation*, consisting of nodes added as well as edges to and from previously committed experience data.

by earlier patches. The data packets associated with the patch updates from the monocular system shown in Figure 3.4 take the form of homographies, with an associated edge representation, $\xi_{homography}$*. This representation is available by estimation of the kind discussed in Section 2.1.2 using sparse feature node representations (e.g.* Υ_{FAST} *and* Υ_{BRIEF}*, as will be discussed in Section 4.3.1 below) extracted from image node representations,* Υ_{image} *– keyframes of which are used to populate the traces at spatially distinct locations (discussed in Section 4.3.2).*

The trading interfaces of Chapter 5 allow for foreign content shared by team members to be effected upon the *graph representation*. Moreover, Chapter 4 tightly integrates the *commutation policy* with the EBN localiser itself. Consequently, these types of patches are heterogeneous in the additive and

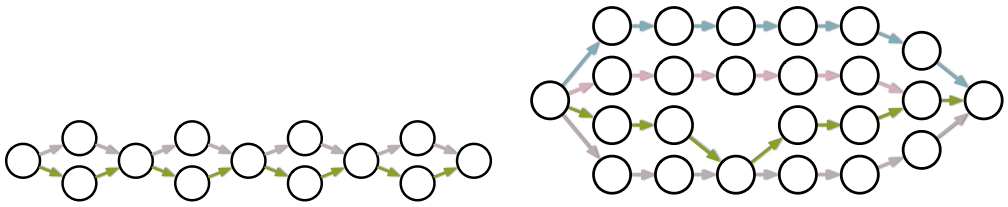
destructive nature of the patch elements which constitute them. Hence, Section 3.3 to follow discusses the asynchronous interaction of agents' *repository representations*.

3.3 Branching Model

This section describes the interaction of the *repository representation* as maintained by multiple agents by virtue of robots engaging in mapping activities during traversals of a common environment as well as via pairwise interactions with other agents in the *team*. The branching model described is appropriate for use in the centralised or distributed trading configurations of Chapter 5 and refers to a generalised *commutation policy*, which is implemented in Chapter 4 and checked for compliance to the model manipulations which follow.

3.3.1 The Local Master Branch

Git manages temporal chronology, i.e. through specifying through its query model "At this point in time the repository consists of this content which was accumulated thus" [Loeliger and McCullough, 2012]. Darcs, in contrast, manages semantic chronology, and specifies through its query model "Currently the repository supports these features and bug fixes" [Roundy, 2005]. Therefore, considering the singular goal of improving localisation, we treat the history of changes, $\mathcal{H}_i$, local to each agent r_i (c.f. Equation 3.6) as a linear sequence, with no support for multiple feature branches and put forward a *history representation* which is more akin to the branching model of



(a) (b)

Figure 3.5: Mutual visualisation of the *history representation* for two distinct teams of robots which are repeatedly mapping the world, and sharing (a) all, or (b) some portion of their database content at every opportunity (the latter of which destroys the symmetry in the local histories as tracked by each robot, except at convergence points). Circles illustrated here represent equivalence in *state representations* (Ξ) rather than database nodes. Divergent patch effects are shown as bifurcating arrows, where the agent's *repository representations* are distinct through the appending of different *patch representations* to their *history representations*, while convergent patch effects are shown as the reconciliation of arrows at convergence basins.

Darcs than that of Git. Figure 3.5 illustrates the interaction of the *history representation* for multiple agents, where convergence points are a result of an equivalence performed on the *state representation* and the pristine state is leftmost in both cases.

3.3.2 Manipulating the Model

Borrowing the nomenclature of the Darcs [Roundy, 2005] community once again, this section phrases DM interactions as "software development" styled actions which agents affect upon the *repository representation* of their currently held content as they engage in the workflow of Figure 1.2 on page 4. Importantly, each agent does not store the *history representation* local to any other agent – only comparing candidate incoming patches to its own sequence of changes. Moreover, in Section 3.4 a scalable demarcation of the

world through its representations as discussed thus far is used to limit the scope of the VCS model and these manipulations.

Init Initialising a pristine *repository representation* (c.f. "start" of workflow, Figure 1.2) involves clearing the *history representation* sequence and appending an empty patch, such that

$$\overset{*}{\Xi}\mathcal{H}_i^{\overset{*}{\Xi}} \leftarrow \{\emptyset\} \tag{3.14}$$

Similarly, the *state representation* is emptied as $\overset{*}{\Xi} \leftarrow \emptyset$. An initial empty patch, $\emptyset$, on the repository history ensures that there is at least one convergence point amongst the *fleet*, which is useful in producing illustrations of their mutual histories (c.f. Figure 3.5).

Record After having embarked on a foray into the world, or midway through a foray, DM agents must commit (c.f. "record", Figure 1.2) the latest changes to their histories, as

$$\overset{*}{\Xi}\mathcal{H}_i^{\overset{a}{\Xi}} \leftarrow \{\overset{*}{\Xi}\mathcal{H}_i^{\overset{*}{\Xi}},\ \overset{*}{\Xi}\mathbf{P}^{\overset{a}{\Xi}}\} \tag{3.15}$$

and the state changes similarly according to Equation 3.13. On the next foray, the patch to appended is computed by inspecting the accumulated history and comparing it to the current database contents

$$\overset{a}{\Xi}\mathbf{P}^{\overset{b}{\Xi}} = \overset{b}{\Xi} \backslash \overset{*}{\Xi}\mathcal{H}_i^{\overset{a}{\Xi}} \tag{3.16}$$

The setwise complement used in Equation 3.16 constructs the patch by

considering all nodes and edges in the updated state that are not already accounted for in the *history representation*.

Diff Section 5.3.3 describes middleware and networking techniques in which patches are communicated between DM agents in centralised or distributed trading configurations (c.f. "request" and "respond", Figure 1.2). Section 3.4.2 describes efficient transformations of the *history representation* before such communication, but for now consider the node and edges from "Experience 1" and "Experience 2" in Figure 3.6 as patches, ${}^{*}_{\Xi}\mathbf{P}^{\overset{b}{\Xi}}$ and ${}^{*}_{\Xi}\mathbf{P}^{\overset{a}{\Xi}}$, summarising respective histories, ${}^{*}_{\Xi}\mathcal{H}_i^{\overset{b}{\Xi}}$ and ${}^{*}_{\Xi}\mathcal{H}_j^{\overset{a}{\Xi}}$, communicated by agents r_i and r_j through these interfaces. Without loss of generality, if the agents' *repository representations* are in states $\overset{a}{\Xi}$ and $\overset{b}{\Xi}$ respectively, the difference in their representations is represented as a virtual repository state, $\overset{u}{\Xi}$, which can be reached by considering the entire residual set of nodes and edges after comparing all those accounted for in both *history representations*.

Tool 3.3.1. ${}^{a}_{\Xi}\mathbf{P}^{\overset{u}{\Xi}} = {}^{*}_{\Xi}\mathbf{P}^{\overset{b}{\Xi}} \backslash {}^{*}_{\Xi}\mathbf{P}^{\overset{a}{\Xi}}$ *and* ${}^{b}_{\Xi}\mathbf{P}^{\overset{u}{\Xi}} = {}^{*}_{\Xi}\mathbf{P}^{\overset{a}{\Xi}} \backslash {}^{*}_{\Xi}\mathbf{P}^{\overset{b}{\Xi}}$ *are the patches that would result in a converged state,* $\overset{u}{\Xi}$*, consisting of all nodes and edges (as encoded in patch elements) not common to the agents involved in the exchange. These are analogous to diff operations in text-based version control, which we refer to as the diff tool.*

At this point, for lightweight communication, these left-hand-side (LHS, c.f. blue shaded areas, Figure 3.6) and right-hand-side (RHS, c.f. purple shaded areas, Figure 3.6) "divergent" patches (${}^{a}_{\Xi}\mathbf{P}^{\overset{u}{\Xi}}$ and ${}^{b}_{\Xi}\mathbf{P}^{\overset{u}{\Xi}}$, respectively) consist of patch elements, p, and as such only node and edge identifiers, v and e. Section 6.2.3 describes how agents sample the relevance of the sensor

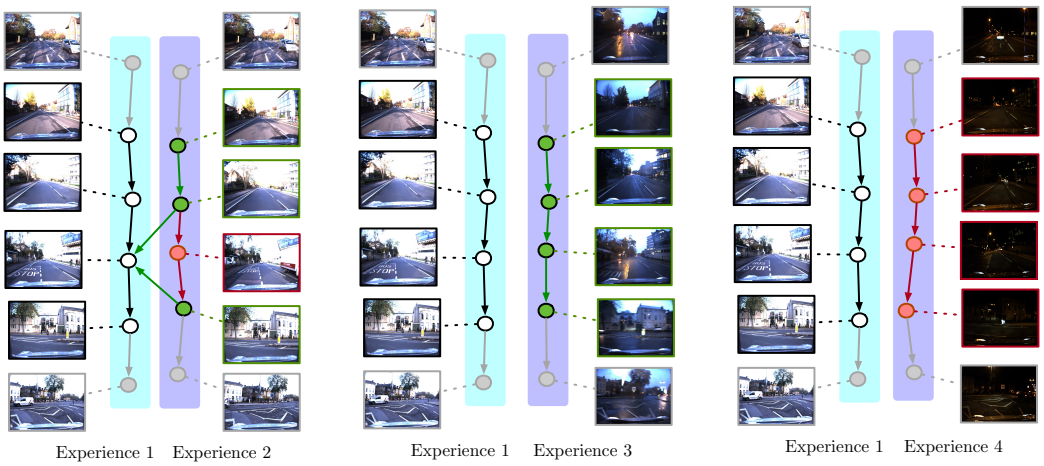


Figure 3.6: Commutation of map data as collected over several experiences of the same place. Here, commutation can be understood as a "splicing" of one experience into another. In some cases (left), two very similar appearances of the same scene location are consolidated by assigning a preferred view to both experiences. Sometimes, an incoming experience cannot be registered to the experience graph at all, and is either permitted into the database structure with no topological links to existing map nodes (middle), or is wholly discarded (right).

data in informing their exchanges, in which the communication of the node and edge representations corresponding to these divergent patches, Υ and ξ, occurs separately and only as necessary.

Pull Section 4.4 tightly couples the behaviour of localisers in lifelong navigation systems to the *commutation policy* in order to produce alternative "convergent" *patch representations* which result in new repository states, $\overset{c}{\Xi}$, which are offset from the virtual state, $\overset{u}{\Xi}$, by considering the redundancy or similarity of sensor data through some portion of the node and edge representations associated with these divergent patches. For now, adapting Equation 3.3, the convergent patches are generally computed for the divergent

patches from Tool 3.3.1 as

$$(\overset{a}{\Xi}\mathbf{P}^{\overset{c}{\Xi}},\ \overset{b}{\Xi}\mathbf{P}^{\overset{c}{\Xi}}) = \Gamma(\overset{a}{\Xi}\mathbf{P}^{\overset{u}{\Xi}},\ \overset{b}{\Xi}\mathbf{P}^{\overset{u}{\Xi}}) \tag{3.17}$$

Here, if the divergent patches are not communicated wholesale and regularly, the size of the patches to be commuted may grow over time, degrading CPU performance. Section 6.1 thus suggests ranking and throttling procedures targeted at content which is likely to register well with their *graph representations*. Additionally, Section 3.4.1 proposes a demarcation of the world into a contiguous set of local attractions and routes.

Section 3.4.2 recasts this versioning procedure as a "purchase" from a responding agent or "seller" to a requesting agent or "buyer" over an itemised data catalogue in preparation for the trading application of Chapter 6, allowing agents to learn which of their *team* members are typically providing higher quality navigation data within those regions in Section 6.2.

Figure 3.6 middle shows the virtual state, $\overset{u}{\Xi}$, as the combination of all nodes and edges in both experiences. Figure 3.6 left shows examples of alternative convergent states, $\overset{c}{\Xi}$, in which LHS and RHS content compete for relevance – where one RHS node is disregarded through the EBN localiser as will be discussed in Section 4.4.

Apply Having commuted divergent *patch representations*, the result is applied to the *graph representation* and *state representation* in a similar fashion to Equations 3.12 and 3.13. However, in contrast to the patches recorded after mapping runs (c.f. Section 3.2.3), these convergent patches may include effects for both adding foreign content and deleting insular content, which

may result in disconnected subgraphs manifesting in the *graph representation*.

Tool 3.3.2. *A tool for incorporating commuted patches while maintaining connectivity in the graph representation, which is referred to as the patching tool,* $\exists$.

The order of this amendment must be approached with care, to avoid corrupting the database schema, and proceeds as follows:

1. **Add nodes**: New nodes (captured in patch elements with actions $\iota = 1$) can be added without any risk of a clash in the *graph representation*, or database schema, so long as all data in the exchanges is uniquely identified, as will be discussed in Section 4.1.2.

2. **Delete edges**: Next, some *edge representations* (oppositely, $\iota = 0$) may be removed, which must occur before the corresponding *node representations* are removed, to avoid breaking the schema.

3. **Add edges**: Then, new edges ($\iota = 1$) may be added only after the nodes which they originate and terminate at have been included (c.f. step 1).

4. **Delete nodes**: Finally, any disregarded insular or foreign nodes may only be forgotten after the edges in the corresponding adjacency lists, æ, have been added (c.f. step 2) and removed (c.f. step 3).

Considering the rightmost scenario shown in Figure 3.6, the *patching tool* as described above will have received as input a candidate convergent

patch, ${}^{\overset{a}{\Xi}}\mathbf{P}^{\overset{c}{\Xi}}$, which includes all of the night-time nodes shown in red. No loop closures are found to the existing content (encoded as the LHS divergent patch, or day-time experience), and the patching procedure discussed above should be modulated to disallow the inclusion of unrelated content, if desired.

Tool 3.3.3. *DM agents may disallow patches to be effected upon the graph representation on an element-by-element basis such that*

$$ {}^{\overset{a}{\Xi}}\mathbf{P}^{\overset{p}{\Xi}} = \{p = (\iota, g),\ g\ =\ (v, æ)\ \in\ {}^{\overset{a}{\Xi}}\mathbf{P}^{\overset{c}{\Xi}},\ \mathbb{E}(\mathcal{G}^{\overset{a}{\Xi}},\ æ) = 1\} \tag{3.18} $$

includes only nodes and edges from the commuted patch which pass a criterion specified by the rollback tool, $\mathbb{E}$*.*

In the simplest case (c.f. Figure 3.6 right), if there is not at least one edge from the incoming foreign content to the insular content already held by the requesting agent, the entire patch is discarded. The *patching tool* and *rollback tool* processes are followed finally by appending the patch which is actually applied to the local *history representation* as

$$ {}^{\overset{*}{\Xi}}\mathcal{H}^{\overset{p}{\Xi}} \leftarrow \{{}^{\overset{*}{\Xi}}\mathcal{H}^{\overset{a}{\Xi}},\ {}^{\overset{a}{\Xi}}\mathbf{P}^{\overset{p}{\Xi}}\} \tag{3.19} $$

and updating the state representation as in Equation 3.13. For a more detailed consideration of the integrity of the *graph representation* under these sorts of commutation routines, refer to Section 4.4.3.

3.3.3 Conflict Resolution

As denoted in the development of the *state representation* and *patch representation* (c.f. Section 3.2.3), we limit the scope of our versioning activities to be discussed in Section 3.3.2 upon the node and edge identifiers which constitute the *graph representation* (which are implemented as fixed-length character strings, c.f. Section 4.3.1). Here, no conflict resolution exists between these versioned identifiers, v and e, and the associated *node representation* and *edge representation*, Υ and ξ, as the EBN implementation of Chapter 4 uses a relational database framework which automatically adds to and prunes the filesystem of binarised sensor logs associated with its entries.

Darcs versus Git Darcs does suffer in comparison to Git when considering large repositories, however, where memory footprints of up to 400 MB when checking in a 50 MB binary file have been reported[3].

Rollback In applying these techniques to an EBN system in Chapter 4, we consider the sensor data relevant to the lifelong navigation problem along with the identifiers in the *graph representation* which are used for versioning. In the strictest case, however, the *patching tool* as described recognises disconnected subgraphs only after attempting to amend the database entries. We thus construct an inverse patch (c.f. Section 3.1) as

$$ {}^{\overset{p}{\Xi}}\mathbf{P}^{\overset{a}{\Xi}} = \{(\neg\iota,\ g),\ p = (\iota,\ g)\ \in\ {}^{\overset{a}{\Xi}}\mathbf{P}^{\overset{p}{\Xi}}\} \tag{3.20} $$

[3] *DVCS Round-Up: One System to Rule Them All? – Part 2*
www.archive.li/7KPXZ#selection-299.0-321.13

and process it through steps 1-4 of the *patching tool* before appending it to the *history representation* as in Equation 3.19 – where nodes added become nodes removed, and vice versa. The *state representation*, likewise, reverts to $\overset{a}{\Xi}$. Section 4.5.3 shows through experimental evidence that the marriage of the *commutation policy* to the EBN localiser – the run-time performance of which dictates how much database content is recorded (c.f. Section 2.2) – ensures that rates of database corruption reach an acceptable steady (albeit fluctuating) state.

3.4 Trading Workflow

This section concludes the chapter by describing a cataloguing of the world as a range of *products* which DM agents use in Section 6.1 to limit the versioning manipulations discussed in Section 3.3 towards mitigating resource shortages (c.f. Section 2.4) as well as for leveraging the relative value that their peers offer to them in a lifelong navigation context.

3.4.1 Product Policies

The grey nodes and edges shown in Figure 3.6 on page 67 illustrate spatial boundaries around an agent's current position in the world which are useful in limiting (a) the load on the *commutation policy* as it uses the EBN localiser in Chapter 4, and (b) the network load during pairwise exchanges in Chapter 5.

Policy 3.4.1. *The product policy, $\wp$, maps database content to unique sec-*

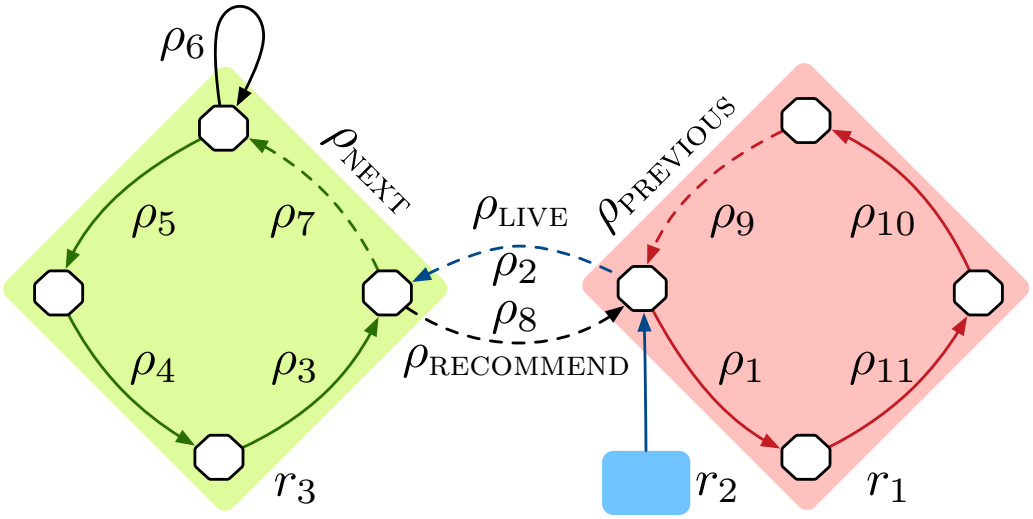


Figure 3.7: The *product policy*, $\wp$, expressed as a route network or *product* space, where each of two robots (r_1 and r_2) are locally expert in a *product* category (red and green, respectively). A third robot (r_3) is a novice. Due to limited hardware and network resources the novice must limit the regularity and partnership of data exchange events and as such needs to approach the choice of trading partner flexibly as it and its peers move around the world.

tions, ρ, of the team's typical operational envelope

$$\wp : v \in \mathcal{G} \mapsto \rho \in \mho \tag{3.21}$$

which (taking into account orientation on traverses) form a catalogue, $\mho$, of non-overlapping products.

Section 3.4.2 uses this framework to frame the versioning activities of DM agents as *vendors* for a range of commodities – each storing a universal topology, implemented as a lookup table, representing sections of the world as well as their connectedness, as shown in Figure 3.7. This topology is distinct from the *graph representation* which each robot is populating by mapping exercises, and in which the metric localisation to be discussed in Section 4.2

is performed.

Catalogue placement An agent targeting its requests (c.f. "request" in the workflow of Figure 1.2) to the appearance of the foray it is currently completing (c.f. Section 6.1) uses the *product policy* for a weak indication of its position within the catalogue.

Tool 3.4.1. *The placement tool,* $\varphi : \Upsilon \mapsto v$*, maps the node representation (c.f. "query node", Figure 3.2) for the query frame through a metric localisation edge (c.f. orange edges, Figure 3.2) to a node identifier already registered to the product policy, which is used as a weak localiser in order to pinpoint agents within the product space,* $\rho_{\text{LIVE}} = \wp(v)$.

The *node representation* is serialised in Section 4.3.1 to contain the navigation data pertinent to this assignment. The number of nodes assigned to each *product* category is not fixed, as agents experiencing the world will continuously assign new content they record, and share these assignments with their peers by virtue of having traded with them.

Segmented experience catalogue To this end, when a dedicated member of the team is treated as a central repository as in Section 5.1 it is able to construct the *product policy* in an online fashion by iterating through the database content submitted by all other agents, mimicking the live data stream that would have been fed to the navigation system by discovering chains (or segments of frames) that are connected linearly by ego-motion – detailed in Algorithm 1. As this process effectively decimates the live camera stream (nodes are recorded by enforcing spatial constraints, as will be dis-

Algorithm 1 In Chapter 4 the database edges are given representations, $\xi_{k-1,k}$, which include incremental motion between nodes, v_k. The *central authority* of Section 5.1, at the point of exchange, has no notion of the experience from which the database content it receives originated. Thus, it incrementally extends linearly connected segments, ρ, of such ego-motion, resulting in the entire environment being eventually treated as a *product* in its own right.

Require: $\xi_{k-1,k},\ \wp : v \to \rho$
Ensure: $\wp : v \to \rho$
$v_{k-1} \leftarrow$ source-node$(\xi_{k-1,k})$ ▷ Correspond segment
$v_k \leftarrow$ target-node$(\xi_{k-1,k})$
$\rho_{k-1} \leftarrow \wp(v_{k-1})$
$\rho_k \leftarrow \wp(v_k)$
if $\rho_{k-1} \wedge \neg\rho_k$ **then** ▷ Prepends segment
    $\wp \leftarrow$ add-to-segment$(v_k,\ \rho_{k-1})$
else if $\neg\rho_{k-1} \wedge \rho_k$ **then** ▷ Appends segment
    $\wp \leftarrow$ add-to-segment$(v_{k-1},\ \rho_k)$
else if $\rho_{k-1} \wedge \rho_k$ **then** ▷ Connects segments
    $\wp \leftarrow$ fuse-segments$(\rho_{k-1},\ \rho_k)$
else if $\neg(\rho_{k-1} \wedge \rho_k)$ **then** ▷ Novel segment
    $\wp \leftarrow$ new-segment$(v_{k-1},\ v_k)$
end if

cussed in Section 4.2), this *product policy* is used in Section 5.4.1 to delegate map building to independent agents, obtaining sizeable maps very rapidly. As the iteratively fused segments of Algorithm 1 will eventually coalesce into a single *product*, the centralised trading applications which follow are suited to small-to-medium scale environments (c.f. Example 3.4.1 below) or offline multi-session map optimisation, experimentally undertaken in Section 5.4.2.

Example 3.4.1. *In the centralised trading applications presented in Chapters 5 and 6 the product,* ρ_{BEGBROKE}*, is named for the* BEGBROKE-24HR *dataset detailed in Appendix A.2.1.*

GPS catalogue In the distributed application of Section 5.3 there is no such *central authority* and the *product policy* is formulated *a priori* for the common environment which agents are operating within. For the purposes of an approximate groundtruth, we have augmented the *Oxford RobotCar Dataset* [Maddern et al., 2017] with a *MATLAB* graphical toolkit for labelling vast collections of video imagery with a direct lookup from frame-index to street name, some samples of which are provided in Appendix A.2.2[4].

Example 3.4.2. *In the distributed trading applications presented in Chapters 5 and 6 products are named according to a local attraction or street, e.g.* ρ_{TRINITY} *refers to the part of the dataset presented in Appendix A.2.2 adjacent to one of the constituent colleges of the University of Oxford, and* $\rho_{\text{BEVINGTON}}$ *corresponds to a suburban street in North Oxford.*

3.4.2 Itemising Data Exchanges

For extensive environments such as described in Example 3.4.2, direct application of the *diff tool* (c.f. Tool 3.3.1) requires the responding agent to communicate a patch update, $\overset{*}{\Xi}\mathbf{P}\overset{b}{\Xi}$, which encompasses its entire *history representation* and will result in the exchange of node and edge identifiers which may already have been communicated between participating agents – aggravating CPU, disk, and network resource limitations as discussed in Section 2.4.

[4]In this thesis, we have created and used labels for approximately 446 km of such data, which we plan to publish at `robotcar-dataset.robots.ox.ac.uk` in a convenient format (e.g. XML, YAML, or JSON).

Category assignment To this end, DM agents maintain various convenient database tables implemented as join clauses through the relational algebra supported by the native database format [Connolly and Begg, 2005] (detailed in Section 4.3.1 by an application of the notation presented in this chapter).

Tool 3.4.2. *The product policy, $\wp$, is used to curate records, $\mathcal{L}_i$, of which nodes belong to each of the distinct regions of the world, ρ_i, as*

$$\mathcal{L}_i = \{v \ \forall \ p = (\iota, g) \in \Xi, \ g = (v, æ), \wp(v) = \rho_i\} \tag{3.22}$$

and which is referred to as the category tool.

There is a direct correspondence between this *category tool* and the *state representation* (the state can be recovered by collecting all patch elements corresponding to the nodes assigned to every *product*) which suggests that this *category tool* alone is not sufficient to limit the size of exchanges patches.

Acquisitions record To this end, in order for requesting agents to further stem the communication of patch elements registered to the *history representation*, DM agents are given the facility to track foreign content that they have received from other members of the *fleet*.

Tool 3.4.3. *The record for foreign content within a product, ρ_k, received from a team member, r_j, consists of the database content for which the node representation is not obtained from raw, unprocessed dataset logs (c.f. Sec-*

tions 2.3.2 and 2.3.3), or

$$\mathcal{A}_{i,j,k} = \{v \ \in \ \mathcal{L}_k, \ \nexists \ \ell \in \mathcal{D}_i \ \cap \ \exists \ \ell \in \mathcal{D}_j, \ \Upsilon \in \ell, \ \Upsilon = \mathcal{G}(v)\} \tag{3.23}$$

and is referred to as the acquisitions tool.

The interpretation of the *history representation* through a trading workflow prevents a requesting agent from repeating the acquisition of database content from another *team* member between data exchange events.

Sales record Conversely, for responding agents to further stem the communication of patch elements registered to the *history representation*, responding agents also avoid offering duplicated database content to another *team* member between data exchange events.

Tool 3.4.4. *DM agents keep track of the insular content within a product, ρ_k, that they have sent to another member of the fleet, r_j, as*

$$\mathcal{S}_{i,j,k} = \{v \ \in \ \mathcal{L}_k, \ v \ \notin \ \mathcal{A}_{i,j,k}\} = \mathcal{L}_k \backslash \mathcal{A}_{i,j,k} \tag{3.24}$$

The collection of such records for all other members of the fleet is referred to as the stock tool.

Accumulation of all such *stock tool* records recovers all nodes in the *state representation* which were not acquired from another agent.

Patch delineation Through the daisy-chained use of the *category tool*, *acquisitions tool*, and *stock tool*, DM agents delineate the *patch representation* communicated to a peer (c.f. "respond" in the workflow of Figure 1.2) based on a query *product*, ρ_{QUERY}. Firstly, the *category tool* is applied:

$$\overset{c}{\Xi}\mathbf{P}^{\overset{d}{\Xi}} = \{p = (\iota, g) \ \in \ \overset{*}{\Xi}\mathbf{P}^{\overset{b}{\Xi}}, g = (v, æ), \wp(v) = \rho_{\text{QUERY}}\} \tag{3.25}$$

This extracts all patch elements in every patch registered to an agent's history which are assigned to the query *product* by the *product policy*. Secondly, the *acquisitions tool* and *stock tool* are applied simultaneously:

$$\overset{e}{\Xi}\mathbf{P}^{\overset{f}{\Xi}} = \{p = (\iota, g) \ \in \ \overset{c}{\Xi}\mathbf{P}^{\overset{d}{\Xi}}, g = (v, æ), v \notin \mathcal{S}_{i,j,k} \ \cap \ v \notin \mathcal{A}_{i,j,k}\} \tag{3.26}$$

The result is a patch which, along with its counterpart as computed by the participating agent become the LHS and RHS *patch representations* versioned through the *diff tool* and *patching tool* mechanisms (c.f. pages 66 to 69). Section 6.1.4 suggests policies for agents to target part of the *product* catalogue based on their current placement within it, ρ_{LIVE}.

Resales DM agents typically – and specifically in all experimental results shown in Sections 5.4 and 6.4 – configured to not "resell" foreign content as retrieved from other agents. However, if so desired, the reselling of foreign content can be explicitly allowed by expanding the scope of the application

of the *acquisitions tool* in Equation 3.26, as

$$ {}^{\frac{e}{\Xi}}\mathbf{P}^{\frac{f}{\Xi}} = \{p = (\iota, g) \ \in \ {}^{\frac{c}{\Xi}}\mathbf{P}^{\frac{d}{\Xi}}, g = (v, æ), v \notin \mathcal{S}_{i,j,k} \cap \ v \notin \mathcal{A}_{i,m,k} \ \forall m \neq i\} \quad (3.27) $$

This prevents a mixed evaluation of content through the pricing strategies which will be introduced in Section 6.2.1 such that DM agents can learn in isolation how typically relevant the experiences of another agent are.

3.5 Summary

In order for later chapters to deal with the implications of the EBN paradigm on the long-term operation of the *fleet*, this chapter has developed the elementary notation this thesis applies to the lifelong navigation problem within the context of MAS, where Section 3.1 introduced basic formalisations of VCS patch theory in terms of the set-theoretic properties of patches applied to repositories of data with non-specific formats.

Section 3.2 introduced the *repository representation* which is maintained atop the traditional map-only representation used in single-agent lifelong navigation systems. Here, Section 3.2.1 described the working state of the repository in terms of database node and edge identifiers which are saved to disk, the specific details of which will be concretely defined in Chapter 4 as we apply the notation developed in this chapter to an existing state-of-the-art EBN implementation. Likewise, Sections 3.2.2 and 3.2.3 explained how the

state of this *repository representation* changes context as the *graph representation* is amended over the lifetime of operation of a single DM agent.

Section 3.3.1 explained the interaction of multiple agents' *repository representations* by introducing the branching model, or *history representation*, which each DM agent stores along with its *graph representation*. The combination of these representations as a complete *repository representation* will be crucial in assuring coherency and consistency in navigation data as held in concert by the team for either the centralised or distributed trading configurations to be discussed in Chapter 5. To this end, Section 3.3.2 discussed several manipulations of the representations put forward in this chapter using the vocabulary typical to VCS operations performed by a software developer.

The versioning metaphors applied thus far consider the trading of experiences over the entire length of the route driven by contributing agents. Thus, Section 3.4.1 demarcated the VCS workflow by considering portions of the environment as content categories to which each agent can contribute, and provided two such *product policies* which will be appropriate for use in centralised and distributed communication networks, respectively. Finally, Section 3.4.2 detailed tools which will be indispensable in Chapter 6 when leveraging the resulting *product policy* for trading targeted at improved localisation under strict resource requirements.

Chapter 4

A Version-Control View of Vision-Based Navigation

This chapter describes an application of the notation set out in Chapter 3 to a state-of-the-art instance of the EBN paradigm which has been applied to single-agent scenarios on vast spatial and temporal scales [Linegar, 2016]. We populate the *node representation* and *edge representation* of the generalised database working copy described in Section 3.2 with sensory records apposite to the lifelong navigation problem which will be leveraged in (a) Chapter 5 for choosing trading configurations which scale well with both the size of the world and the size of the *team* in terms of the endless accumulation of both insular and foreign database content, and specifically (b) Section 5.3.3 by designing DM communication tools to mitigate the elevated costs of transmitting these data packets, and (c) Chapter 6 for targeting efficient exchange events or branching model manipulations (c.f. Section 3.3) which are likely to boost localisation performance. Indeed, this chapter focuses on the conse-

quences of the VCS interpretation which simultaneously impact and enhance the behaviour of the EBN localisation module, which is framed as the *canonical DM client* going forward by its inherent interpretation of localisation as a metric graph relationship. Foreseeing trouble due to the not necessarily additive nature of the *patching tool* effects after commutation (c.f. "Deleting nodes" and "Deleting edges", item 2 and item 4 on page 69), we advance the notation of test configurations and test cases as a level of guarantee that candidate *commutation policies* maintain the integrity of the *graph representation*. To this end, we show the conformance of this experience mapper to the requirements set out in Chapter 3 by using it to explore both theoretical and practical considerations when implementing a valid *commutation policy* through a testing and simulation framework which gives strong guarantees of the consistent health of the *graph representation*, insights which are reaffirmed through experiments with modestly sized teams of cooperating agents.

4.1 Graphs

This section discusses the population of elements of the generalised working copy of the database schema discussed in Chapter 3 via *node representations* and *edge representations* which are important for lifelong navigation clients participating in the DM.

4.1.1 Characteristics of Place and Motion

This section draws on [Hartley and Zisserman, 2003] and the notation of [Ma, 2004] to provide an overview of aspects of multiple view visual geometry which are relevant to this work.

Image Formation Cameras project 3D points, $\mathbf{X} = (x, y, z)^T \in \mathbb{R}^3$, an example of which is show in Figure 4.1(a), onto a sensor plane, $\mathbf{x} = (u, v)^T \in \Omega$, to form an image, $\mathcal{I}$. Here, the image is considered as a map for the 2D photographic medium, Ω, to an array of brightness values, $\mathbb{R}^+$, such that

$$\mathcal{I} : \Omega \subset \mathbb{R}^2 \rightarrow \mathbb{R}^+ \tag{4.1}$$

The values taken by the map, $\mathcal{I}$, are discretised (e.g. for greyscale images where $\mathbb{R}^+ \sim [0, 255]$) and depend upon physical properties of the scene being viewed, including: shape, material reflectance, and the configuration of sources of illumination. As illustrated along the principal axis in Figure 4.1(b), image pixels in $\mathbb{R}^2$ can be represented in a projective space $\mathbb{P}^2$ using a homogeneous co-ordinate system, $\bar{\mathbf{x}} = (u, v, 1)^T$. Similarly, spatial locations in the scene can be represented in projective space $\mathbb{P}^3$ as $\bar{\mathbf{X}} = (x, y, z, 1)^T$. As indicated in Figure 4.1(a), images captured consecutively (at time steps $k-1$ and then k, where $t(k-1)$ and $t(k)$ are typically separated by the framerate) are denoted as $\mathcal{I}_{k-1}$ and $\mathcal{I}_k$.

Camera behaviour is dominated by both internal parameters and their configuration with respect to other sensors mounted on field robots.

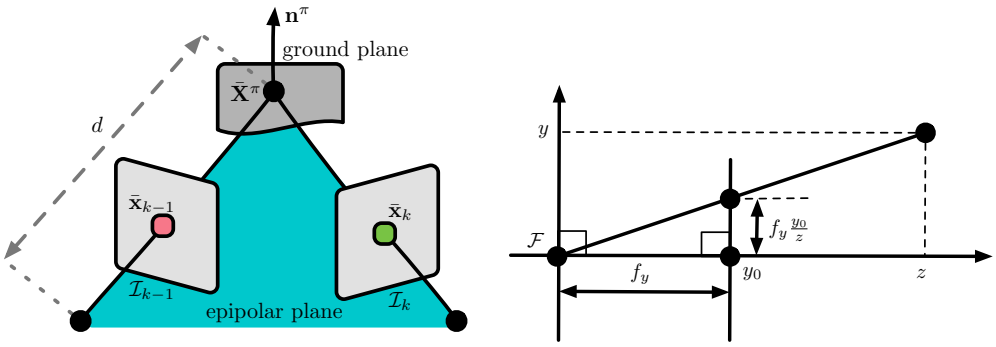


Figure 4.1: (a) We exemplify scene structure by a point, $\bar{\mathbf{X}}^\pi$, situated on the ground plane, π, that is parametrised by its normal, $\mathbf{n}^\pi$, and the perpendicular distance to the centre of the first camera view, d. (b) A pin-hole camera with centre of projection, $\mathcal{F}$, and the principal axis parallel to the z axis. The image plane is at focus and hence focal length f_y away from the centre of projection.

Intrinsic Camera Models The camera intrinsics are encoded in a matrix, $\mathbf{K} \in \mathbb{R}^{3\times3}$, which combines the focal lengths, f_x and f_y, principal points, x_0 and y_0, and a skew distortion parameter, s, as

$$\mathbf{K} = \begin{bmatrix} f_x & s & x_0 \\ 0 & f_y & y_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.2}$$

Here, f_x and f_y are in units of pixels, and are distinct if the image is formed from pixels which are not square. The distortion factor, s, is typically negligible but gains relevance when the sensor frames of reference are not orthogonal. The principal offsets, x_0 and y_0, describe the alignment between sensor plane and camera centre.

Correction of radial distortion In cameras with radial distortion, straight lines in space are in general images as curves. In pixel coordinates the correction is accomplished by a non-linear function which is typically computed by minimising a cost based on the deviation from a linear mapping [Hartley and Zisserman, 2003].

Extrinsic calibration of sensor suites These camera intrinsics may be calibrated once and assumed to be constant using toolboxes such as that presented in [Scaramuzza et al., 2006]. However, while not explicitly addressed in this thesis, mobile robots are commonly equipped with a range of equipment and sensors, such that methods that automatically estimate the extrinsic calibration between sensors from natural scenes are indispensable [Scott et al., 2016]. Moreover, calibration parameters can be maintained over the lifetime of a robot by automatic methods which learn to detect that parameters are no longer viable or correct [Furgale et al., 2015].

Projections Now, points in the scene and their corresponding image pixels, when expressed in homogeneous co-ordinates, are related by $\bar{\mathbf{x}} = \mathbf{PT}\bar{\mathbf{X}}$, where $\mathbf{P} \in \mathbb{R}^{3\times4}$ is a projection matrix which relates the relative position and orientation of the camera's and world's frames of references, as expressed in the six degree-of-freedom (6DoF) transformation, $\mathbf{T}$, of the kind that will be discussed below. The projection matrix is related to characteristics of the camera's hardware as $\mathbf{P} = \mathbf{K}[\mathbf{I}|\mathbf{0}]$. If the relative position and orientation of the camera's and world's frames of references is known, $\mathbf{T}$ may be factored out and the resulting projection matrix refers to a *normalised camera*.

Lie Groups A point in the scene in two frames of reference, $\bar{\mathbf{X}}_A$ and $\bar{\mathbf{X}}_B$, can be related by a rigid-body transformation, $\mathbf{T}_{A,B} \in \mathbb{SE}(3)$, such that $\bar{\mathbf{X}}_A = \mathbf{T}_{A,B}\bar{\mathbf{X}}_B$. The homogeneous transformation matrix is composed of a rotation, $\mathbf{R}_{A,B} \in \mathbb{SO}(3)$, and translation, $\mathbf{t}_{A,B} \in \mathbb{R}^3$, such that

$$\mathbf{T}_{A,B} = \begin{bmatrix} \mathbf{R}_{A,B} & \mathbf{t}_{A,B} \\ \mathbf{0} & 1 \end{bmatrix} \tag{4.3}$$

In general, rotations are parameterised by a sequence of twists around mutually orthogonal axes in the frame of reference (x, y, and z), corresponding firstly to a yawing motion, $\phi \in [-\pi, \pi]$, around z such that

$$\mathbf{R}_z(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix} \tag{4.4}$$

followed by a pitch, $\beta \in [-\frac{\pi}{2}, \frac{\pi}{2}]$, around y such that

$$\mathbf{R}_y(\beta) = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix} \tag{4.5}$$

and finally a roll, $\alpha \in [-\pi, \pi]$, around x such that

$$\mathbf{R}_x(\alpha) = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.6}$$

and ultimately obtain the total rotation as

$$\mathbf{R} = \mathbf{R}_z(\phi)\mathbf{R}_y(\beta)\mathbf{R}_x(\alpha) \tag{4.7}$$

where we are following the North-East-Down (NED) NASA convention [Cai et al., 2011].

Epipolar Geometry Here we describe the intrinsic projective geometry which relates two overlapping camera views, which in the case of field robots navigating with vision correspond to subsequent frames in video imagery or adjacent frames from a binocular stereo head.

The Fundamental Matrix The Fundamental Matrix, $\mathbf{F} \in \mathbb{R}^{3\times3}$, encapsulates the intrinsic projective geometry between two overlapping views, $\bar{\mathbf{x}}_k$ and $\bar{\mathbf{x}}_{k-1}$, of a point in the world, $\bar{\mathbf{X}}$, such that

$$\bar{\mathbf{x}}_{k-1}^T \mathbf{F} \bar{\mathbf{x}}_k = \bar{\mathbf{x}}_k^T \mathbf{F} \bar{\mathbf{x}}_{k-1} = 0 \tag{4.8}$$

where the intuition is as indicated in Figure 4.1(a), such that rays from each of $\bar{\mathbf{x}}_k$ and $\bar{\mathbf{x}}_{k-1}$ to a 3D point, $\bar{\mathbf{X}}^\pi$ (which in Figure 4.1(a) happens to be situated on the ground plane), are traced out on the epipolar plane and constrain $\bar{\mathbf{x}}_k$ and $\bar{\mathbf{x}}_{k-1}$ to lie on their respective epipolar lines: where the epipolar and image planes coincide.

As discussed above, if the camera can be assumed to be calibrated, then imaged points are expressed in "normalised coordinates" – with respect to the camera's coordinate frame. The Fundamental Matrix corresponding

to a pair of normalised cameras is referred to as the "Essential Matrix" [Hartley and Zisserman, 2003].

4.1.2 Graph Tools

Requiring the *node representation* and *edge representation* to be populated with sensory and processed data in this way, we facilitate further in Section 5.3.3 the communication of navigation information between processes. The data packets associated with the *edge representation* will typically be much smaller than their corresponding *node representations*, especially if the VO implementation does not cache (or save to disk) the 3D landmarks after extraction.

Packet formats The data packets for these *node representations* and *edge representations* consist of XML headers and a binary blob containing a serialised Google Protocol Buffer[1], being suited to both local storage and the middleware communication services of Chapter 5 [Sumaray and Makki, 2012].

Relational database In this chapter, these packets are retrieved locally by *postgres* [Stonebraker and Rowe, 1986] or *sqlite* [Owens and Allen, 2010] SQL queries. The unlimited scope of spatio-temporally indexed data as collected by a *fleet* of autonomous vehicles and entered into the DM requires a departure from typical *ad hoc* approaches to the collection and storage of experiences. We use in this thesis the relational database and data management framework of [Nelson et al., 2015] which distinguishes clearly between

[1] `https://developers.google.com/protocol-buffers/docs/reference/overview`

metadata and sensor data.

Unique data identification To this end, we use 128 bit Universally Unique Identifiers (UUIDs) [Leach et al., 2005] to annotate each database node, v, and edge, e. This ensures the ability to track the lifetime usage of data (be it images, laser scans, etc.) used in experiences as they are passed between robots and processes, and also ensures that no duplication of frames takes place. This is indispensable when the sharing of navigation data is instigated by *ad hoc*, pairwise interactions – for example in the centralised trading application of Section 5.1 when preventing an agent that has downloaded a frame from trying to re-upload to the *central authority*.

Search Seeding pose estimation over wide baselines (c.f. Section 4.2 below) requires that the *edge representation* facilitates composition over the space of its relational operator.

Tool 4.1.1. *The edge representation can be concatenated as*

$$ {}^{i}\xi_{n} \;=\; {}^{i}\xi_{j} \;\oplus\; {}^{j}\xi_{k} \;\oplus\; \ldots \;\oplus\; {}^{m}\xi_{n} \tag{4.9} $$

along a path of connected places radiating outwards from a query node, v_i, which is discovered by the search tool, Ψ, up to a maximum spatial extent, d_{max}, or $\{{}^{i}e_{j},\ {}^{j}e_{k},\ \ldots,\ {}^{m}e_{n}\} = \Psi(\mathcal{G},\ v_{i},\ d_{max})$.

For the 6DoF EBN *edge representation* (c.f. Section 4.3.1), compositions take the form of matrix multiplications in the $\mathbb{SE}(3)$ group. When composing the *edge representation* with the distance metric of Equation 4.16, the

triangle inequality

$$|^i\xi_j| + |^j\xi_k| \geq |^i\xi_k| \tag{4.10}$$

is a requirement for optimisation in a metric space [Khamsi and Kirk, 2011]. The depth limit, $d_{\max}$, is typically set to a 25 m radius circle around the robot's current location.

Optimisation However, these compositions are not required to be accurate for farflung i and n as we require only locally accurate place-to-place relations, except in the case that the search limits originate from nodes assigned by the *category tool* (c.f. Section 3.4.2) through the *product policy* (c.f. Section 3.4.1) to *products* which are adjacent in the route network of Figure 3.7. Section 6.1.4 suggests trading strategies for targeting data exchanges between *products* which are regularly related by the EBN system in this way. The *graph representation* can be uni- or bi-directional in practice, to facilitate ease in implementation of this *search tool*. This design choice is informed by complexity considerations, where a min-priority queue implementation of Dijkstra's search will scale linearly with the number of edges, $O(|E| + V \log V)$ [Dijkstra, 1959]. EBN *graph representation* traversals are implemented in this thesis as breadth-first searches [Bundy and Wallen, 1984].

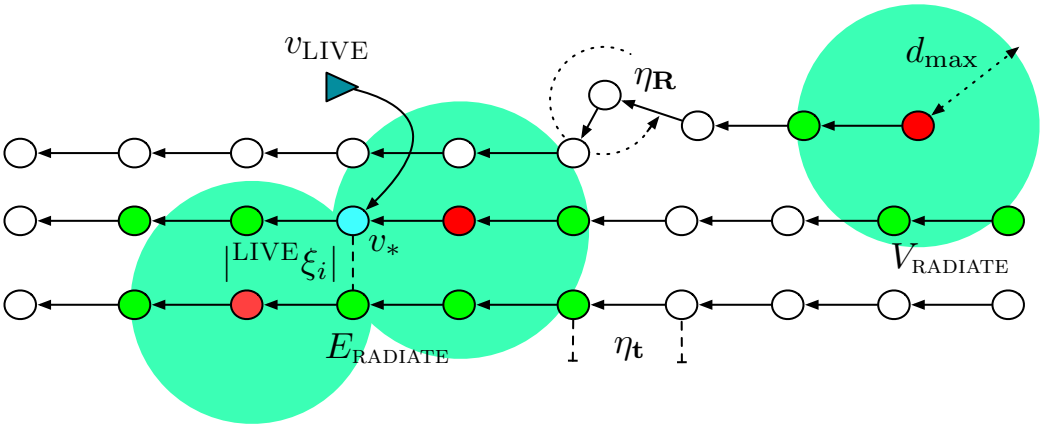


Figure 4.2: Seed estimates (shown red) from FAB-MAP provide priors on the robots pose in the experience graph. Candidate nodes are then retrieved by performing limited searches through the topometry surrounding the seeds (green shaded areas).

4.2 Localisation as a Metric Graph Relationship

As shown in Figure 4.2, EBN localisation within the topometric experience graph is limited to optimisation within these local graph neighbourhoods which radiate outwards from nodes identified by place recognition or loop-closing techniques. This restricted approach is efficient and is expedited by recent localisation success. Furthermore, the localiser's run-time performance dictates the amendment of the underlying *graph representation* of the world.

Automatic initialisation Localisation begins by selection of a set of candidate frames which closely match the query, $\Upsilon_{\text{FAB-MAP}}(v_{\text{LIVE}}) = \mathbf{b}_{\text{LIVE}}$, in appearance using FAB-MAP (c.f. Section 2.1.3), yielding a candidate node set, $V_{\text{SEEDS}} = \mathcal{L}_{\text{FAB-MAP}}(\mathcal{G},\ v_{\text{LIVE}})$, consisting of topological seeds (shown red in Figure 4.2) which may include false positives (incorrect data associations).

Traversing the topometry The *graph representation* is then traversed radiating outwards (c.f. Figure 4.2, green shaded area) from each of the topological seeds

$$E_{\text{RADIATE}} = \{e \in \Psi(\mathcal{G},\ v_i,\ d_{\max}) \forall v_i \in V_{\text{SEEDS}}\} \tag{4.11}$$

This results in a coverage over the *graph representation* of candidate node sets (c.f. Figure 4.2, green nodes)

$$V_{\text{RADIATE}} = \{v_i : {}^{i}e_j \in E_{\text{RADIATE}} \cup {}^{j}e_i \in E_{\text{RADIATE}}\} \tag{4.12}$$

as well as an *edge representation* (shown dashed in Figure 4.2) accumulated through callbacks to the *search tool*

$$|{}^{\text{LIVE}}\xi_i| = |{}^{\text{LIVE}}\xi_j||{}^{j}\xi_k|\ldots|{}^{m}\xi_i|,\ v_i \in V_{\text{RADIATE}} \tag{4.13}$$

as it builds paths of edges from the live frame to each node in the radiated set (c.f. Equation 4.9).

Wide baseline correction Ensuring features have a similar viewpoint in frames of comparison provides an additional level of protection against spurious matching between frames that are captured some time apart, and not in an incremental fashion – beneficial to detectors which are not robust to significant viewpoint variation (c.f. Section 2.1.1). Figure 4.3 shows how these baselines can be reduced using *edge representations* which are homographies induced by planes in the scene (which are related to the motion undergone by the robot, c.f. Section 2.1.2). Local landmarks in the query image plane

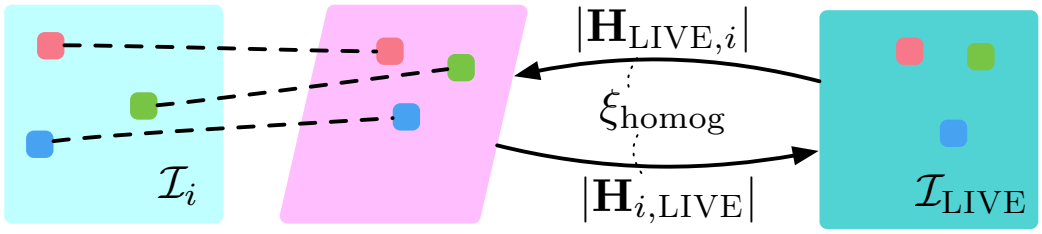


Figure 4.3: Guiding the final pose estimate by expressing local sparse features stored in the query *node representation* in the frame of reference for the candidate node.

(coloured square regions, right) are repositioned (pink, middle) by applying the accumulated search estimate, $|^{\text{LIVE}}\xi_i|$, allowing feature matching (dashed lines, c.f. page 18) to better find correspondences in the candidate image (light blue, left), $\Upsilon_{\text{image}}(v_i) = \mathcal{I}_i$, which are finally translated back into to the query frame of reference before proceeding with pose refinement.

Geometric refinement LSE pose refinement (c.f. Section 2.1.1) from the query to each of these candidates is solved with GN techniques, as LM may fail to converge if initially far from a minimum, resulting in the most promising node for localisation (blue node in Figure 4.2)

$$v_* = \underset{v_i \in V_{\text{RADIATE}}}{\operatorname{argmin}} |^{\text{LIVE}}\xi_i| \tag{4.14}$$

However, we impose a strict requirement on bonafide localisation results such that they must satisfy further geometric constraints. A typical value of 3.00 px is used for the RANSAC inlier threshold of Equation 2.3. An average reprojection error of 0.41 px typically leads to a 5 % speed estimation offset for passenger vehicles [Buczko and Willert, 2016].

Spatial ordering To avoid recording novel *node representations* in the degenerate case of a stationary robot, EBN only adds nodes and edges to the graph when the translation between two keyframes exceeds a threshold, $\eta_{\mathbf{t}}$ (typically 0.5 m). A similar threshold, $\eta_{\mathbf{R}}$, is applied to the rotation the robot undergoes (typically 45° or 0.79 rad). Spatially ordering the map by applying these thresholds (shown in Figure 4.2) allows the robot to repeat trajectories (c.f. Section 2.2.2) at arbitrary speeds.

Plastic mapping As described in [Churchill and Newman, 2012b], if localisation is sufficiently good, the robot is considered to hold a satisfactory representation of the part of the world that it is currently mapping, and no new *node representations* or *edge representations* are recorded. Specifically, independent experiences are each run as a localiser in parallel and $N = 3$ is taken as the minimum acceptable agreement count.

Enriching related experiences However, if the query *node representation* has been added to the *graph representation*, and localisation is sufficiently good, the localisation result is used to amend the graph such that an edge is created between it and the query node. It should be clear that when localisation is not sufficiently good, it may prove erroneous to add query frames to the *graph representation*, as over the course of a foray a robot may never localise, and will be left with disconnected subgraphs in its map (which would have the incremental effect of degrading all further searches over the *graph representation*). Section 4.5.3 shows experimentally that the localiser effectively "mends" the *graph representation* on subsequent forays by match-

ing in appearance space to these disjoint experiences, thereby creating paths which the *search tool* can traverse [Churchill and Newman, 2012a].

Failure recovery Therefore it is critical that each candidate passes a multitude of tests to prevent erroneous amendments of the *graph representation*. EBN localisation as described, being divorced from a probabilistic framework and lacking introspection, occasionally stalls in the face of so many checks for robustness – resulting in failures which were characterised in Section 2.3.1. In order to avoid missing too many opportunities to create meaningful edges in the map (problematic if the robot has travelled blindly for long enough such that the maximum depth of the *search tool* prevents it from ever recovering) we perform at this point of failure another very modest topological search over the set of map nodes which can be obtained from a constant velocity model. This contingency need only be resorted to in the case of extreme lens glare or unmodelled robot kinematics. Upon recovery, localisation proceeds in a well-behaved manner.

Prioritising experience Without intelligent predictions about which memories to next consider, an unmanageable list of experiences (to search and update) manifests itself and imposes unreasonable requirements on the disk space available to field robots. Hence, much literature tries to prune such ever-growing *graph representations* to a manageable level, either by limiting spatial density as in [Milford and Wyeth, 2010], aggregation of cumulative visual experience through topics [Paul et al., 2012], or the inclusion of only landmarks that are most useful for place recognition [Dymczyk et al., 2015].

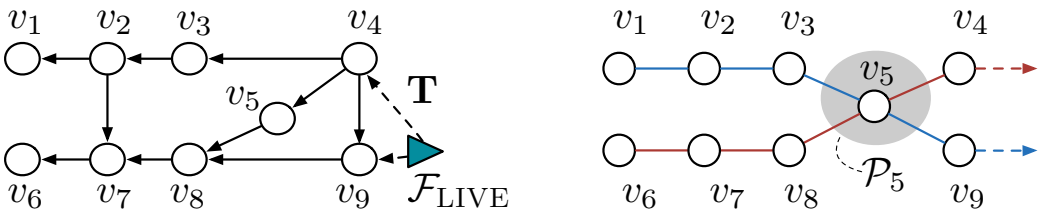


Figure 4.4: "Experience Graphs" are implemented as a database of nodes, v, connected by directed edges, which define topometric neighbourhoods linked by 6DoF relative pose, $\mathbf{T}$, expressed in this figure in a robot-centric frame of reference, $\mathcal{F}_{\text{LIVE}}$. Adjacent topologies, such as the "Path Memory" network (right) recording the history of localisation success [Linegar et al., 2015] are not accounted for in distributed DM applications, but are exploited for correlating experience by a *central authority* in Section 6.1.2.

4.3 Confirming Representations

This section shows conformance of EBN to the *graph representation* (c.f. Section 4.3.1) and as such introduces the *canonical DM client* as the lifelong navigation system used by DM clients for the remainder of the thesis. Additionally, the *patch representation* is expounded as an induced subgraph of the *graph representation* in which graph tools can be used to express metric graph relationships between the data-bearing *node representations* and leveraged to mimic the localisation process during pairwise exchanges.

4.3.1 The Experience Graph

The topometric "Experience Graph" used by EBN and shown in Figure 4.4 (left) only differs from the generalised database working copy of Figure 3.2 in its omission of the mirror (or transpose graph) collection of reversed edges [Harary, 2005], and as such is a member of the class of database schemas which this thesis treats as the VCS working copy.

Characteristics of space Indeed, conflating the "Experience Graph" with the *graph representation* requires that the *edge representation* of Equation 3.8 on page 58 must be uniquely and consistently invertible (i.e. ${}^{i}\xi_{j} = {}^{j}\xi_{i}^{-1}$). Due to EBN's use of stereo VO (c.f. Section 2.1.1) this *edge representation* takes the form of 6DoF rigid-body transformations, $\xi_{\text{6DoF}}(e) = \mathbf{T}$, which are indeed invertible as per Equation 2.1

$$\begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix}^{-1} = \begin{bmatrix} \mathbf{R}^T & -\mathbf{R}^T\mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} \tag{4.15}$$

The *graph representation* is flexible to this characterisation, however, and may take the form of planar-induced homographies (c.f. Section 2.1.2).

Parameterising rotation The mapping of the Euler rotational components in Equation 4.4 to Equation 4.6 on page 87 into Euclidean space must be approached with care to avoid gimbal lock [Stuelpnagel, 1964]. EBN avoids this by an over-parameterisation using quaternions [Hamilton, 1844] and the associated *edge representation*, $\xi_{\mathbb{SO}(3)}(e) = \mathbf{q} \in \mathbb{R}^4$. For both localisation and commutation (c.f. Sections 4.2 and 4.4.1) a measure of spatial "closeness" between two related places is then available using a non-negative norm of the *edge representation*

$$|{}^{i}\xi_{j}| = |{}^{j}\xi_{i}| \geq 0 \tag{4.16}$$

which in this thesis is ignores orientation offset and simplifies to the physical distance $\xi_{\text{translation}}(e) = \mathbf{t} \in \mathbb{R}^3$.

Image characteristics To estimate these elements of ego-motion from 3D landmarks (c.f. Section 2.1.1) EBN requires several variations of the *node representation* of Equation 3.7 on page 58 including keypoints, $\Upsilon^j_{\text{FAST}}(v) = \bar{\mathbf{x}}^j$, and their corresponding descriptor patterns, $\Upsilon^j_{\text{BRIEF}}(v) = \mathbf{d}^j$. These features are extracted from stereo image pairs consisting of two views with purely horizontal parallax [Seitz and Kim, 2002] which are bundled together as $\Upsilon_{\text{image}}(v) = \mathcal{I}$ and which share (to sub-millisecond accuracy) an image capture time $\Upsilon_{\text{tod}}(v) = t$ through the clock synchronisation method of [Harrison and Newman, 2011].

Epipolar calibration During fleetwide sharing of information, this estimation is assumed to be general to the intrinsic properties of incoming frames due to an exchange of images between distinctly equipped and configured agents. This assumption means that in the current implementation, as we will discuss further in Section 5.3.3, entire images are sent over the network only after rectification. The required calibration parameters are applied as a Fundamental Matrix (c.f. Section 4.1.1).

Characterising place In contrast to these point-feature methods for rectified video sequences, loop closures (c.f. Section 2.1.3) are identified using the SURF detector and descriptor to maintain an alternative *node representation*, $\Upsilon^j_{\text{SURF}}(v) = (\bar{\mathbf{x}}^j,\ \mathbf{d}^j)$. A trained FAB-MAP model, along with the *product policy* (c.f. Section 3.4.1), is held by each agent and is not subject to versioning. Measuring visual similarity or "closeness" in appearance for two views of the same place, as Section 6.2.1 requires for pricing data transac-

tions, the global image descriptor *node representation*, $\Upsilon_{\text{BoW}}(v) = \mathbf{b}$, yields the bag of words used internally by FAB-MAP.

Example 4.3.1. *The patch representation, carrying no sensor logs (c.f. Representation 3.2.7), of the multicolor experience graph shown in Figure 3.4 on page 62 is identical regardless of the algorithm that records it.*

Thus it is only through coupling the *node representation* and *edge representation* to the specific EBN system that DM agents can guide the model manipulations of Section 3.3.2.

4.3.2 Stereo Localisation in the Experience Graph

This section provides some detail on how EBN accomplishes localisation in topometric graph structures as discussed generally in Section 4.2.

Metric localisation RANSAC relies on a minimum number of data points needed to estimate model parameters, m, which in the perspective from three points (P3P) case imposes $m \geq 3$ [Fischler and Bolles, 1981] and the winning transformation is that with the greatest number of data associations in agreement with $\mathbf{T}_{k-1,k}$. RANSAC ensures that this transformation is computed by samples from the inlier set alone with a probability p if the number of trials adheres to the log ratio in Equation 2.2, where e is the probability of selecting an inlier from the set of associations, and is impacted by the signal noise in the image sensor.

FAB-MAP EBN leverages an open-source FAB-MAP implementation which proves to be a bottleneck in performance due to the looking up of a vo-

cabulary match across all nodes in the database (linearly degrading in the database size) [Glover et al., 2012]. Chapter 6 shows how this can be mitigated in a centralised framework presented in by a *download policy* which considers only mission-critical database content, and in the fully distributed case in Chapter 5 by a discriminative selection of trading partner, a choice which is guided in Chapter 6 by a more nuanced understanding of the local expertise and value that other agents provide in distinct sections of the world.

"Path Memory" The history and connectedness of successful localisations form a topology atop the topometric poses called "Path Memory", $\mathcal{P}$, a term introduced in [Linegar, 2016], which effectively relates *node representations* that represent a view of the environment under similar conditions. Some such paths are shown in Figure 4.4, identified by UUIDs (c.f. Section 4.1.2), and associated to the working copy by a *node representation*, $\Upsilon_{\text{paths}}(v)$, which consists of every path which touches a node, v (a subset of the entirety of paths registered to "Path Memory", $\mathcal{P}$). Chapter 5 leverages this historical co-localisation of the *graph representation* to select statistically relevant sequences of nodes to download.

4.4 Confirming Darcs Attributes

Consider that the *patch representation* can be thought of as an induced subgraph as it is formed from a subset of the nodes of the *graph representation* and all of the edges connecting pairs of those nodes [Diestel, 2005] This

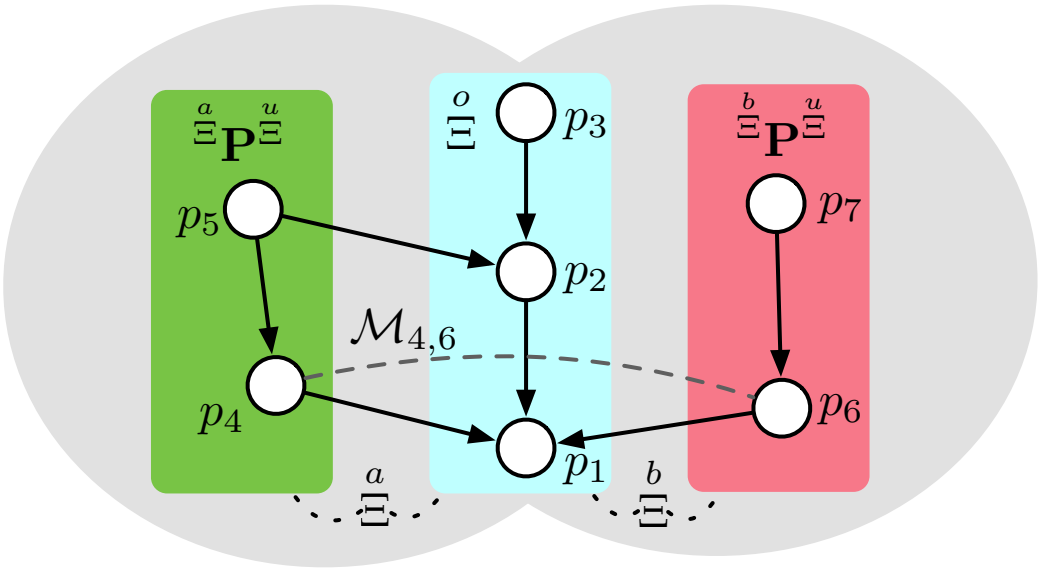


Figure 4.5: A pairwise exchange of patches by *canonical DM client* agents with some common repository history (blue), but currently divergent map representations (green and red). To guarantee symmetry and consistency in such an exchange, and particularly over the lifetime of a *fleet* of arbitrary size, agents must share a common understanding of how to commute divergent content as well as agree on a common protocol for valuing, choosing between, and discarding from closely matching content (dashed).

section thus describes the seamless application of localisation routines between the *graph representation* and *patch representation*, the latter of which can be assembled from basic patch elements as in Figure 3.3 and shown alongside the corresponding database content as in Figure 4.2 and further.

4.4.1 Incorporating Foreign Experience

EBN DM clients use the *diff tool* (c.f. Tool 3.3.1) in conjunction with a globally unique identification scheme (c.f. Section 4.1.2) to ensure that localisation effort as expressed through the *commutation policy* is best spent on matching divergent content.

Example 4.4.1. *Figure 4.5 illustrates the states of two robots' repositories,* $\mathcal{R}_i^{\overset{a}{\Xi}}$ *and* $\mathcal{R}_j^{\overset{b}{\Xi}}$*, at the point of initiating a trade. The repository representations share an original common state* $\overset{o}{\Xi} = \{p_1, p_2, p_3\}$ *(they have traded with each other before) but (through either trading with other agents or by virtue of having mapped the world recently) have diverged in their representations such that* $\overset{a}{\Xi} = \{p_1, p_2, p_3, p_4, p_5\}$ *and* $\overset{b}{\Xi} = \{p_1, p_2, p_3, p_6, p_7\}$*. Consider two candidate convergent patches,* ${}^{\overset{b}{\Xi}}\mathbf{P}^{\overset{u}{\Xi}}$ *(containing the patch elements* $\{p_4, p_5\}$*) and* ${}^{\overset{a}{\Xi}}\mathbf{P}^{\overset{u}{\Xi}}$ *(likewise,* $\{p_6, p_7\}$*), which would transform each repository to a common state:* $\overset{u}{\Xi} = \{p_1, \ldots, p_7\}$*.*

Union commute Example 4.4.1 illustrates the simplest strategy for commuting divergent patches where agents include all foreign content which is not already accounted for in their *graph representation*. Providing detail for non-specific commutation of the data to be versioned (c.f. Policy 3.1.1) of Equation 3.17 (c.f. page 68)

$$({}^{\overset{a}{\Xi}}\mathbf{P}^{\overset{c}{\Xi}},\ {}^{\overset{b}{\Xi}}\mathbf{P}^{\overset{c}{\Xi}}) \ = \ \Gamma_{\cup}({}^{\overset{b}{\Xi}}\mathbf{P}^{\overset{u}{\Xi}},\ {}^{\overset{a}{\Xi}}\mathbf{P}^{\overset{u}{\Xi}}) = ({}^{\overset{b}{\Xi}}\mathbf{P}^{\overset{u}{\Xi}},\ {}^{\overset{a}{\Xi}}\mathbf{P}^{\overset{u}{\Xi}}) \tag{4.17}$$

The validity of $\Gamma_{\cup}$ can be verified in considering the effect of these patch updates on the *history representations*, ${}^{\overset{*}{\Xi}}\mathcal{H}^{\overset{c}{\Xi}} \leftarrow \{{}^{\overset{*}{\Xi}}\mathcal{H}^{\overset{a}{\Xi}},\ {}^{\overset{a}{\Xi}}\mathbf{P}^{\overset{c}{\Xi}}\}$ and ${}^{\overset{*}{\Xi}}\mathcal{H}^{\overset{c}{\Xi}} \leftarrow \{{}^{\overset{*}{\Xi}}\mathcal{H}^{\overset{b}{\Xi}},\ {}^{\overset{b}{\Xi}}\mathbf{P}^{\overset{c}{\Xi}}\}$, as per Equation 3.19. Indeed, the *state representation* that this *commutation policy* targets, $\overset{c}{\Xi} = \overset{u}{\Xi}$, may include content in each agent's map which can be sufficiently explained by existing *node representations*, aggravating the ever-increasing experience density along with the associated bottlenecks in performance (c.f. Section 2.4), in conflict with the EBN paradigm (c.f. Section 4.2, "plastic mapping"). Additionally, wholesale manipulation

of the branching model in this way does not take into account the *state representations* associated with paying attention to the health of the *graph representation* by the *patching tool* or *rollback tool* (c.f. Section 3.3.2).

Localiser commute To this end, the *commutation policy* detailed in Algorithm 2, Γ_{darcs}, and used throughout Chapters 5 and 6 employs the EBN localisation module itself to search for the set of pairwise matches between the divergent *patch representations*

$$\mathcal{M} \;=\; \{(p_i,\; p_j) \mid p_i \;\in\; {}^{\overset{b}{\Xi}}\mathbf{P}^{\overset{u}{\Xi}},\; p_j \;\in\; {}^{\overset{a}{\Xi}}\mathbf{P}^{\overset{u}{\Xi}}\} \tag{4.18}$$

consisting of all pairs of LHS and RHS nodes with *node representations* which are relatable through the topometry (c.f. Figure 4.3) and which pass the geometric refinement steps discussed above. Disregarding closely matching content in this way is expressed through versioning operations by keeping insular (LHS) content, and ignoring the matched foreign (RHS) content, or vice versa. Each of these considerations can be problematic in their own way – we discuss the related issues in Sections 4.4.2 and 4.4.3 to follow. Figure 4.5 shows one such pairwise match (dashed line).

4.4.2 Maintaining Coherency in Pairwise Exchanges

Indeed, in order to maintain symmetry in the *graph representations* of various agents in the *fleet*, Γ_{darcs} relies on a repeatable and universal (to the team) method for deciding which of two matched nodes should be retained and which should be removed from the *graph representation*, emerging on line 17

Algorithm 2 This *commutation policy* uses the *canonical DM client* to disregard redundant patch content and pays heed to the *rollback tool*. Divergent patches as input and the convergent patch which the requesting agent obtains are as described in Figure 3.5 (page 64). Here, although *node representations* for RHS content are required (line 13) to perform the localisation routines, the RHS *graph representation* is – in the results presented in this chapter – hosted on the same physical hardware as that of the LHS. Section 5.3 details DM middleware technology which allows the communication of these data packets, which is also useful for implementing the *choice tool*.

Require: ${}^{a}_{\Xi}\mathbf{P}^{u}_{\Xi}$, ${}^{b}_{\Xi}\mathbf{P}^{u}_{\Xi}$, $\mathcal{G}$ ▷ Divergent patches, LHS *graph representation*

Ensure: ${}^{a}_{\Xi}\mathbf{P}^{p}_{\Xi}$, $\mathcal{M}$ ▷ LHS convergent patch, pairwise matches

1: ${}^{a}_{\Xi}\mathbf{P}^{p}_{\Xi} \leftarrow \emptyset$, ${}^{b}_{\Xi}\mathbf{P}^{p}_{\Xi} \leftarrow \emptyset$, $\mathcal{M} \leftarrow \emptyset$
2: **for** $p = (\iota,\ g)\ \in\ {}^{b}_{\Xi}\mathbf{P}^{u}_{\Xi}$ **do** ▷ Clear LHS of RHS deletions
3: **if** $\iota = 0$ **then**
4: ${}^{a}_{\Xi}\mathbf{P}^{p}_{\Xi}\ \leftarrow\ {}^{a}_{\Xi}\mathbf{P}^{p}_{\Xi} \cup p$
5: **end if**
6: **end for**
7: **for** $p = (\iota,\ g) \in\ {}^{a}_{\Xi}\mathbf{P}^{u}_{\Xi}$ **do** ▷ Clear RHS of LHS deletions
8: **if** $\iota = 0$ **then**
9: ${}^{b}_{\Xi}\mathbf{P}^{p}_{\Xi}\ \leftarrow\ {}^{b}_{\Xi}\mathbf{P}^{p}_{\Xi} \cup p$
10: **end if**
11: **end for**
12: **for** $p = (\iota,\ g)\ \in\ {}^{b}_{\Xi}\mathbf{P}^{u}_{\Xi},\ g = (v,\ æ)$ **do** ▷ RHS added content
13: $v_*\ \leftarrow$ EBN-localiser$(v,\ \mathcal{G})$ ▷ Localise to LHS
14: $æ_* \leftarrow$ get-edges$(\mathcal{G}, v_*)$
15: **if** $p_* = (1, g_*), g_* = (v_*, æ_*) \in\ {}^{a}_{\Xi}\mathbf{P}^{u}_{\Xi}$ **then** ▷ Good localisation
16: $\mathcal{M} \leftarrow \mathcal{M} \cup (p_*,\ p)$
17: **if** $\mathbb{E}(\mathcal{G}, æ)$ **then** ▷ Check *rollback tool*
18: $v_+ \leftarrow \gamma(v_*,\ v)$ ▷ Choose LHS/RHS
19: **if** $v_+\ \neq\ v_*$ **then** ▷ Replace LHS with RHS
20: ${}^{a}_{\Xi}\mathbf{P}^{p}_{\Xi}\ \leftarrow\ {}^{a}_{\Xi}\mathbf{P}^{p}_{\Xi}\ \cup\ (0, g_*)$
21: ${}^{a}_{\Xi}\mathbf{P}^{p}_{\Xi}\ \leftarrow\ {}^{a}_{\Xi}\mathbf{P}^{p}_{\Xi}\ \cup\ p$
22: **end if**
23: **end if**
24: **else** ▷ No localisation, add to LHS
25: ${}^{a}_{\Xi}\mathbf{P}^{p}_{\Xi}\ \leftarrow\ {}^{a}_{\Xi}\mathbf{P}^{p}_{\Xi}\ \cup\ p$
26: **end if**
27: **end for**

in the implementation detail (c.f. Algorithm 2).

Health of the *graph representation* Considering the effect of choices of this kind on the database schema, we must ensure (if the *rollback tool* dictates as much) that the inclusion of "deleting" patch elements (c.f. line 20, v_-, $\iota = 0$) within the convergent *patch representation* maintains connectivity in the local graph neighbourhood, i.e. no disconnected subgraphs result from the effect of the *commutation policy*. Some choice mechanisms which would clearly cause mischief of this kind include (a) consistent selection of LHS content, or (b) random selection based on a coin toss.

Tool 4.4.1. *DM agents make use of a consistent and invertible choice tool for deciding which of two pairwise matching elements,* $v_+ \leftarrow \gamma(v_i, v_j)$*, should be preferred and which should be ignored,* $v_- \leftarrow \gamma^{-1}(v_i, v_j)$*. Additionally, indecision for frames which do not closely match is allowed and denoted as* $v_+ = v_- = v_?$ *such that* $\gamma(v_i, v_j) \in \{v_i, v_j, v_?\}$*, where* $v_?$ *indicates (by control of some threshold by whichever process is using the choice tool) dissimilarity in the frames and is implemented as an invalid UUID character string.*

However, aside from the purity of the VCS interpretation, this *choice tool* is inherently related to the quality of the *graph representation* and as such to the performance of the EBN system. Chapter 6 use a *choice tool* implementation which consider FAB-MAP compatibility of frames in advance of running the entire EBN localisation routine in order to rank and throttle data exchanges and filter for frames which are likely to localise well to one another.

Quality of representation Indeed, [MacTavish et al., 2017] demonstrate safe and constant-time VT&R on a 12 km route over a 31 h period by promoting image similarity between the live view and relevant experiences based on what the vehicle is currently seeing, without having to consider all possible environmental influences on scene appearance. Sections 6.4.1, 6.4.2 and 6.4.6 show experimentally that image-based comparisons can be useful as heuristics for expected localisation performance. Unless stated otherwise, the *choice tool* considers the average score returned by a FAB-MAP query over the *node representations* assigned by the *category tool* (c.f. Section 3.4.2) to the *product* targeted for exchange, ρ_{QUERY} (c.f. Section 3.4.1).

Sharing representations Chapter 5 details the exchange of the *node representation* data packets which this *choice tool* requires. Clearly, searching for all possible pairwise matches (c.f. Algorithm 2) requires every RHS *node representation* to be communicated, which conflicts with the resource management concerns of Section 2.4. Section 6.1.3 thus presents *throttling* and *ranking* mechanisms to assuage the network and CPU load associated with this *commutation policy*.

Example 4.4.2. *One possible use of the choice tool may leverage the "Path Memory" statistics (c.f. Section 4.2)*

$$\gamma(v_i, v_j) = \begin{cases} v_i, & \textit{if } |\Upsilon_{paths}(v_i)| > |\Upsilon_{paths}(v_j)| \\ v_j, & \textit{otherwise} \end{cases} \tag{4.19}$$

only if the node representations, Υ_{paths}, for both insular and foreign content are available.

This chapter thus explores the VCS interpretations of Chapter 3 through studying the implications of this *commutation policy* for simulated and real MAS, members of which are considered separate processes with shared physical connections.

4.4.3 Maintaining Map Integrity Under Patch Updates

This localiser-inspired *commutation policy* must do additional work to maintain the integrity of the experience graph by avoiding corruption in the form of inconsistent or missing directed edges between pairs of nodes, which can be catastrophic for motion planners in VT&R scenarios (c.f. Section 2.2.2). This section thus concludes the chapter by proposing tools for investigating the behaviour of any *commutation policy* and dictates strict requirements on the layout of arbitrary pairs of divergent patches when considering a fixed battery of designed test conditions.

Test configurations Consider any arbitrary collection of divergent pairs of *patch representations*, $\mathcal{C} = \{\mathcal{C}_i = ({}^{\overset{o}{\Xi}}\mathbf{P}^{\overset{a}{\Xi}},\ {}^{\overset{o}{\Xi}}\mathbf{P}^{\overset{b}{\Xi}})\}$.

Example 4.4.3. *The test configurations of Figure 4.6 (left) are each designed to include a single pair of LHS and RHS node representations which would, upon exchange between DM agents, have been matched by the EBN localisation module (c.f. Equation 4.18).*

These configurations must be designed such that, under the action of the *commutation policy*, the divergent *patch representations* they carry can commute to a sensible *graph representation*.

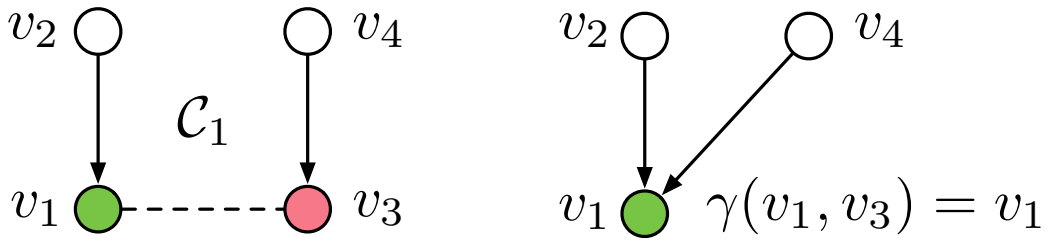
Example 4.4.4. *Figure 4.6 shows (right) the graph representation which the application (c.f. patching tool, Section 3.3.2) of the LHS and RHS convergent patches must result in as they account for the chosen (green) and disregarded (red) nodes as dictated by the choice tool.*

The conditions for testing these scenarios should in general be designed to check that the node chosen by the *choice tool* maintains connectivity with all of the adjacent nodes of its disregarded counterpart, for which the *search tool* (c.f. Tool 4.1.1) is indispensable.

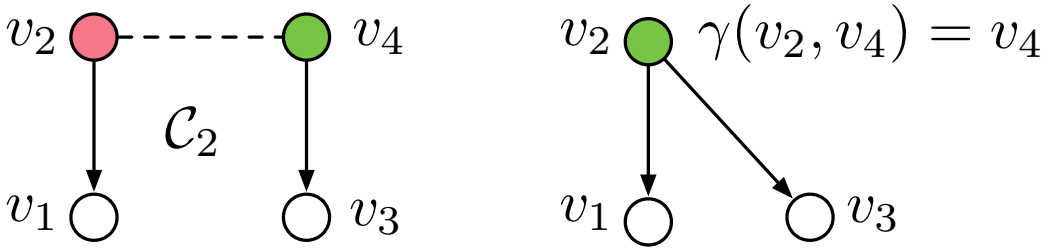
Test batteries Thus the designer must specify a battery of such test conditions, $\mathcal{T} = \{\mathcal{T}_j\}$, each of which indicates the correctness of the *commutation policy* in terms of its ultimate effect on the degrees of freedom for each node represented in a test configuration, $\mathcal{T}_j(v) \in \{0, 1\}$.

Example 4.4.5. *Figure 4.6 uses a test battery which consists of two test conditions – the first of which considers whether there are any nodes in the configuration which are involved in a match (e.g. v_3 top, v_2 middle, v_2 and v_3 bottom), and the second of which considers whether there are any nodes in the configuration which are joined by an edge to a node which is involved in a match (e.g. v_2 to v_1 top).*

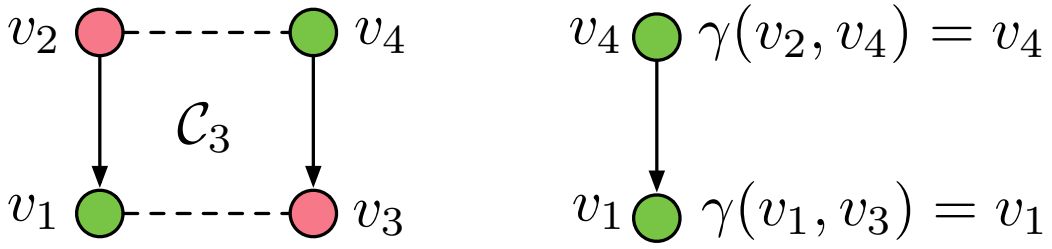
Test vectors The ("yes" or "no") result of the battery on a particular node yields a binary test-vector over that node, $\mathcal{T}(v) = [\mathcal{T}_1(v), \mathcal{T}_2(v), \ldots]^T$. This yields a multiset of test-vectors, $\mathcal{T}(\mathcal{C}_i) = \{\mathcal{T}(v_1), \mathcal{T}(v_2) \ldots\}$, when the battery is run over the entire configuration.



(a) This first configuration fails to fully satisfy the test battery as there is no node that both points to another node that is involved in a match, and is involved in a match itself.



(b) Similarly, this second configuration fails as there is no node that is chosen from and connected to another node that is chosen from.



(c) The third test alone will also fail to test the battery satisfactorily, as there is now no node that is neither chosen from, nor points to a node that is chosen from.

Figure 4.6: Three candidate configurations of divergent patches for verifying the integrity of the *graph representation* under the action of the *commutation policy*. Divergent patches are shown left, and the expected converged *graph representation* is shown right in each case. A *commutation policy* which non-exhaustively applies the *choice tool* (i.e. request the associated *node representation* data packets for incoming foreign content) may omit the redirection of edges coinciding with the disregarded content to the preferred content (deleting the edge from v_4 to v_1 in (a) results in v_4 occupying its own subgraph).

Example 4.4.6. *The first configuration, $\mathcal{C}_1$, shown in Figure 4.6(a) results in a multiset of test-vectors, $\mathcal{T}(\mathcal{C}_1) = \{[1,0]^T, [0,1]^T, [1,0]^T, [0,1]^T\}$, where each of the test cases is satisfied, but no part of the configuration tests both cases in tandem (i.e. there is no node in this configuration which is both chosen from and connected in the topometry to a node that is chosen from).*

Example 4.4.7. *The second test configuration, $\mathcal{C}_2$, shown in Figure 4.6(b), similarly fails, $\mathcal{T}(\mathcal{C}_2) = \{[0,0]^T, [1,0]^T, [0,0]^T, [1,0]^T\}$, to test the entire battery. Here, $\mathcal{T}(v_1) = \mathcal{T}(v_2) = [0,0]^T$, indicates that nodes v_1 and v_2 do not fall within the purview of the test battery (c.f. Example 4.4.5).*

Cardinality checks All tests evaluated over all nodes in every candidate configuration yields the multiset union $\mathcal{T}(\mathcal{C}) = \bigcup_{\mathcal{C}_i \in \mathcal{C}} \mathcal{T}(\mathcal{C}_i)$. If the cardinality of the support for this multiset satisfies $\text{card}(\text{supp}(\mathcal{T}(\mathcal{C}))) = 2^{|\mathcal{T}|}$ then the collection, $\mathcal{C}$, is considered to have sufficiently tested the battery, $\mathcal{T}$.

Example 4.4.8. *It is only through the introduction of a third test configuration, shown in Figure 4.6(c), that the multiset of test-vectors over all configurations, $\mathcal{T}(\mathcal{C}) = \{[0,0]^T, [0,1]^T, [1,0]^T, [1,1]^T\}$, achieves cardinality $2^2 = 4$.*

Commutation stress test simulator It is beyond the scope of this work to suggest an irrefutable test battery or enumerate every corner-case collection of divergent patches, and as such these test configurations may be randomly generated or may be parsed from exemplar mapping data from EBN outputs. Section 4.5.1 presents results using realistic experience maps which show that the *commutation policy* of Algorithm 2 occasionally intro-

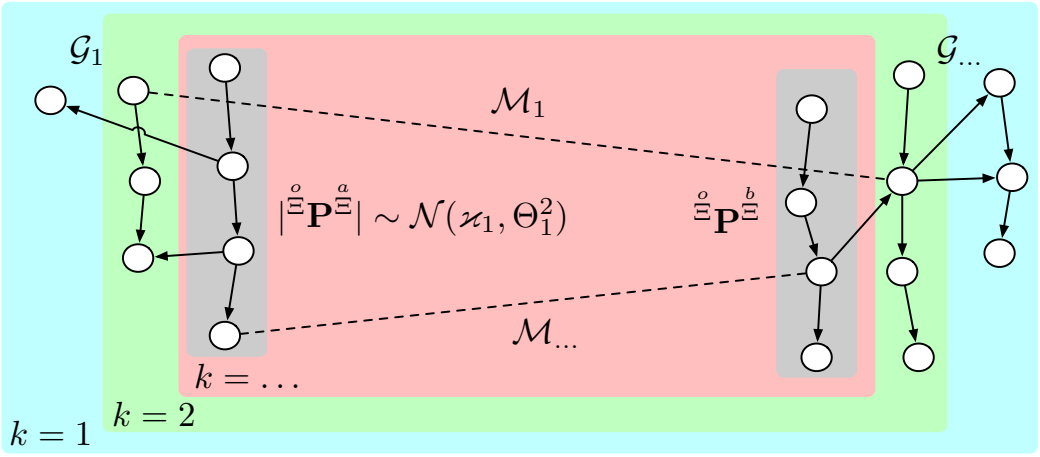


Figure 4.7: Generative simulator to verify the integrity of the *graph representation* of various virtual agents under the action of the *commutation policy*. At each iteration (shaded blue, green, red, respectively), each agent (two of which are shown here) produces a variable number of new nodes for its map. Upon exchanging patches (grey) a set of randomised pairwise matches (dashed, black) are specified consistently between the processes that represent each agent's repository. Edges to alternative parts of the *product* space (c.f. Section 3.4.1) are not supported by this *skeleton tool*, which should not impact the convergence guarantees in terms of node counts, but may induce a residual difference in edges as tracked by the working copy.

duces corruption in the form of disconnected subgraphs. To this end, Figure 4.7 illustrates a simulation scheme, which (when repeated often enough) presents the *commutation policy* with randomised test configurations which capture all corner-cases in database connectivity.

Tool 4.4.2. *The skeleton tool shown in Figure 4.7 generates (for DM agents r_i) random patch representations with node counts distributed normally as $\mathcal{N}(\varkappa_i, \Theta_i^2)$. This ensures that (at the point of trading on simulation step k) the expected node count in each robot's graph representation is $k\sum \varkappa_i \pm k\sum \Theta_i^2$.*

As the *commutation policy* is wed to the localiser, subsequent visits to an area of the world, should in effect "mend" the working copy by match-

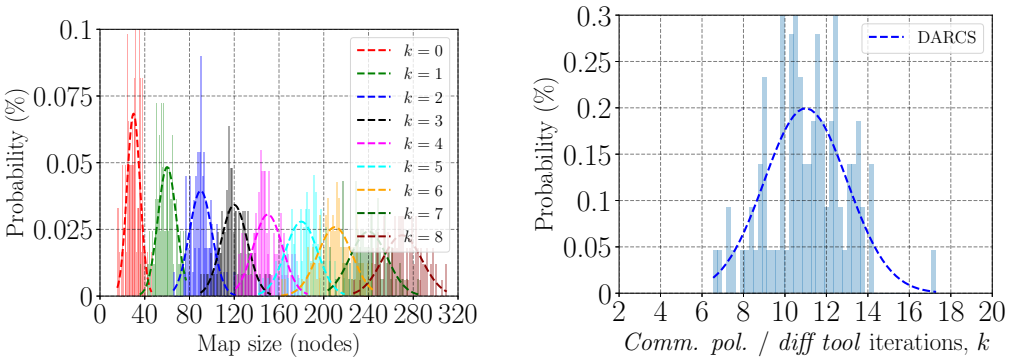
ing in appearance space and adding topometric constraints between disjoint parts of the *graph representation*. Sections 5.3.1 and 5.3.2 extend this *skeleton tool* by handshaking and flow control mechanisms for MAS simulations between many networked machines.

4.5 Results

This section tests the validity of the VCS interpretation of Chapter 3 as applied to the EBN DM client instantiated in this chapter. In particular, we are concerned with the system's convergence properties when applying a *commutation policy* which is designed to leverage the EBN localisation module (c.f. Section 4.4.1) as opposed to a wholesale inclusion of all foreign content. Additionally, realistic sources of inconsistencies introduced to the working copy representations (c.f. Section 3.2.1) by incorporating metric graph relationships (c.f. Section 4.2) within the model manipulations of Section 3.3.2 are presented and shown to be manageable.

These reasonably strong guarantees for DM agents always being able to consolidate divergent representations of the same environment (c.f. Section 1.2) are accompanied by a discussion of the implication for disk use (c.f. Section 2.4), anticipating the approaches for managing these concerns in Chapters 5 and 6.

The results which follow, with the exception of Section 4.5.1, were obtained by aggregating statistics over randomised trials with no held-out sets (c.f. Section 2.3.3), and a *fleet* consisting of two agents operating on the same physical machine (c.f. Section 4.4.2). Video sequences are sourced from



(a) (b)

Figure 4.8: Distribution of (a) node counts at each convergence basin, $1 < k < 9$, with lengthening tails at later stages in the simulation, where the average size of the *graph representation* goes from 30 ± 6 nodes to 270 ± 14 nodes, and (b) number of data exchanges required before the *fleet* achieves convergence after an initial divergent mapping run.

five of the *product* space regions specified by the GPS catalogue *product policy* (c.f. Section 3.4.1) over the *Oxford RobotCar Dataset* discussed in Appendix A.2.2.

4.5.1 Commutation Policy Verification

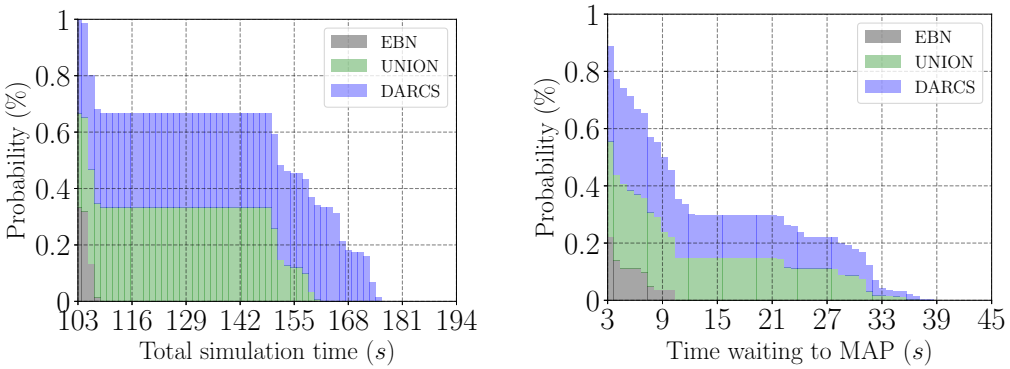
To investigate the appropriateness of the VCS formalisation (item 1 on page 5), this experiment continues the discussion of Example 4.4.8 to Example 4.4.3 by using the *skeleton tool* to provide the *commutation policy*, Γ_{darcs}, with randomised *patch representations* for consolidation. Specifically, a *fleet* of two robots engage in nine forays, each of which begins with the recording of toy *graph representations* and concludes in a wholesale trade of the entire difference in representations (c.f. *diff tool*, Tool 3.3.1). The simulation parameters (c.f. Figure 4.7) are set to $\varkappa_1 = 20$, $\Theta_1 = 5$ and

$\varkappa_2 = 10,\ \Theta_2 = 3$ for each agent, respectively.

Figure 4.8(a) shows the average size of the team's *graph representations* (by node count) at nine convergence points (i.e. data exchange events) which follow the mutual *history representations* originally shown in Figure 3.5(a) on page 64, over the course of one hundred iterations of mapping and trading. Here, it is evident that trading with other robots introduces uncertainty into the insular *graph representation*, and that without effective strategies for managing data exchanges (discussed in Chapter 5), maps will grow unreasonably.

In contrast to this scenario, Figure 3.5(a) on page 64 illustrates *history representations* which do not regularly agree, but ultimately converge. The intermediate discrepancy in representation may be due to *throttling* and *ranking* of data exchanges (to discussed in Section 6.1.3), targeting of demarcated regions of the world (discussed in Section 6.1.4), or canny selection of trading partner (c.f. Section 6.2.3). For now, however, agents respond to each other's requests with patches of size 3 ± 1 and Figure 4.8(b) shows the normally distributed expected trial length before convergence after an initial generation of toy mapping data (again 20 ± 5 and 10 ± 3 nodes for each agent, respectively), taking 11 ± 2 trials on average.

While Section 4.5.3 explores the behaviour of the system in terms of the *edge representations* from a disturbance-rejection perspective, it is clear from this experiment (in which no trials failed to converge) that the *commutation policy* as wed to the EBN localisation module provides strong guarantees of convergence in terms of *node representations*.



(a) (b)

Figure 4.9: Cumulative distributions of (a) total simulation times and (b) time spent waiting to transition in *flow representation* for a team of two robots using various *commutation policies* to register incoming patches against simultaneously enlarging maps over fifty trials. In (b), the Darcs *commutation policy* spends more time in associating incoming frames with the current state of the *graph representation*, where the probability of experiencing a delay longer than 15 s is 29.63 % (DARCS, blue) as compared to 14.81 % (UNION, green). Similarly, in (a), a total simulation time of 120 s is 66.67 % likely (DARCS, blue) as compared to 33.33 % (UNION, green).

4.5.2 Convergence Properties

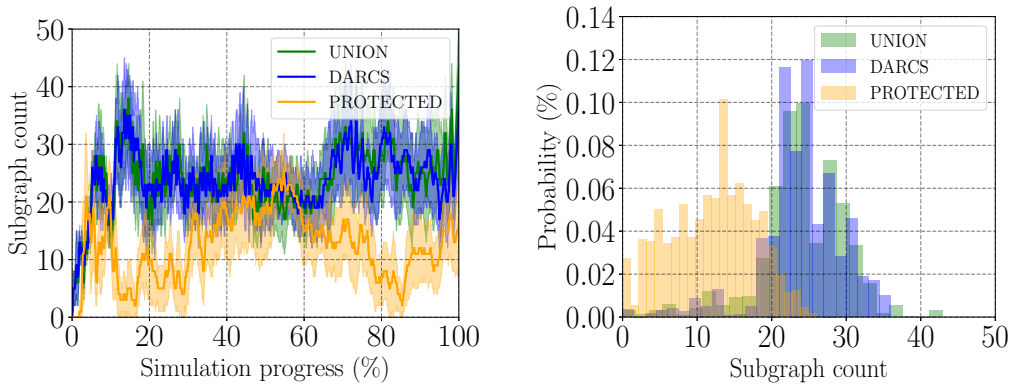
To realistically explore convergence properties of the system when the divergence in representation is not sampled from a normal distribution (c.f. Section 4.5.1), this experiment departs from the use of simulated toy data through the *skeleton tool* and aggregates fifty trials in which two agents engage in five forays over a 20 m portion of the ρ_{BROAD} *product*.

Figure 4.9(a) shows a distribution of the total time taken to completely map and exchange all experiences, while Figure 4.9(b) shows the lag time typically experienced by agents between mapping runs as a measure of how much processing time is spent on applying the branching model manipulations of Chapter 3. Section 5.3.1 presents a *flow representation* which is used as an extension to the *skeleton tool* and which is used to take measurements for Figure 4.9(b).

Here, the cumulative distributions present the probability that the duration of the simulation or activity is in excess of a query time. For comparison, Figure 4.9 also shows timing results for single-agent EBN (no data exchanges are performed).

Here, the Darcs *commutation policy* induces extra effort on the CPU, with average (total) simulation times of $169.27\,\text{s} \pm 3.93\,\text{s}$ and average versioning times of $14.2\,\text{s} \pm 0.35\,\text{s}$ as compared to the Union *commutation policy* (c.f. Section 4.4.1, $153.07\,\text{s} \pm 3.8\,\text{s}$ and $11.93\,\text{s} \pm 0.34\,\text{s}$, respectively).

While both *commutation policies* ultimately mandate an exchange of *node representation* and *edge representation* data packets (c.f. Section 4.4.2), the additional processing load induced by Γ_{darcs} is attributed to an exhaustive



(a) (b)

Figure 4.10: Corruption of the database schema expressed as (a) accumulated subgraphs as the simulation proceeds and (b) histogram of corruption rates. The marriage of localiser and *commutation policy* along with an approximating block on irrelevant incoming content (c.f. Tool 3.3.3, *rollback tool*) provides some robustness to drift. Here, the VCS approach to lifelong navigation tends to recover from mistakes, through the addition of metric localisation edges by the *canonical DM client* itself.

search for pairwise correspondences (c.f. Algorithm 2, line 12). This effect aggravates the guiding question for scalable operation (item 2 on page 5) and is managed in Chapter 6 by a discriminatory choice of trading partner based on a lightweight sampling of *node representation* which requesting agents can use as an early rejection mechanism for data exchanges which are not likely to correspond via the EBN localisation module.

4.5.3 Rejecting Graph Representation Disturbances

The simulated convergence statistics of Sections 4.5.1 and 4.5.2 do not concern themselves with the health of the *graph representation* as expressed through *edge representation* connectivity. Figure 4.10(a) thus aggregates a cumulative count of the number of subgraphs present in *graph representa-*

	PUSEY	REWLEY
UNION	$17.70\,\% \pm 3.04\,\%$	$20.55\,\% \pm 3.70\,\%$
DARCS	$14.25\,\% \pm 3.12\,\%$	$19.16\,\% \pm 6.57\,\%$
PROTECTED(100 %)	$52.80\,\% \pm 6.18\,\%$	$41.70\,\% \pm 7.62\,\%$

Table 4.1: Corruption rates expressed as the fraction of data exchange events at which the *graph representation* is within an acceptable threshold of perfect (fewer than ten subgraphs, lower is better). The *commutation policies* are 26.99 % and 45.95 % (DARCS) or 33.52 % and 49.28 % (UNION) as effective without including the baseline protection offered by the *rollback tool* in the PUSEY and REWLEY *product* spaces, respectively.

tions for two agents which perform ten randomised trials of forty five forays, each over a 20 m portion of the $\rho_{\text{ST-ANNES}}$ *product*. Here, decimal accuracy is a result of interpolating the sampled counts over many trials.

Application of the *commutation policy* of Algorithm 2 without the *rollback tool* is seen to induce a heavier error rate (blue peaks are slightly offset from green peaks) as compared to the wholesale commutation of $\Gamma_{\cup}$, which does not perform view-deletion of closely matching views by the *choice tool* (c.f. Tool 4.4.1). Indeed, deploying the *rollback tool* to anticipate and protect the map against the inclusion of imagery which does not match to insular *graph representation* content results in a lower bound on the error rate (orange troughs). Figure 4.10(b) shows an accompanying histogram for these counts, where the combined *rollback tool* and Darcs *commutation policy* exhibit an average error rate of 13 ± 6 subgraphs as compared to the Darcs and Union *commutation policies* without such protection (24 ± 6 and 25 ± 6, respectively).

At about 50 % of the average trial, the *rollback tool* approach tends to experience equivalent corruption rates as the unprotected approaches. Con-

sidering Equation 3.18 on page 70, this can be attributed to the approximate nature of *rollback tool* decision on a per-element basis, failing to consider connectivity amongst nodes in the incoming *patch representation* itself. In all three cases, the steady-state error rate settles to an acceptable level (in the sense that the time series are stationary). This is attributed to the EBN localisation module itself, where subsequent forays have a "mending" effect as each robot locates itself and creates topometric links between experiences, vindicating this chapter's coupling of *commutation policy* and localiser.

Table 4.1 presents error rates which are measured as the fraction of data exchange events at which the condition of the *graph representation* is within an acceptable threshold of perfect (set to ten subgraphs). In these supplementary experiments, two agents perform ten randomised trials of twenty three forays, each over a 40 m portion of the ρ_{PUSEY} and ρ_{REWLEY} *products*, respectively. The combination of Darcs *commutation policy* and *rollback tool* once again results in the highest fidelity of *edge representation*.

4.5.4 Commuting and Inducing Forgetfulness

Considering as guiding questions item 3 and item 4 on page 5, this experiment explores the impact of expressing the EBN framework through versioning operations on the optimality of data exchanges as well as localisation outlook regardless of the inevitable inconsistencies experimentally indicated in Section 4.5.3. Here, two agents perform ten randomised trials of forty five forays, each over a 20 m portion of the $\rho_{\text{ST-ANNES}}$ *product*.

The Darcs *commutation policy* exhibits similar impact on the disk (c.f. Sec-

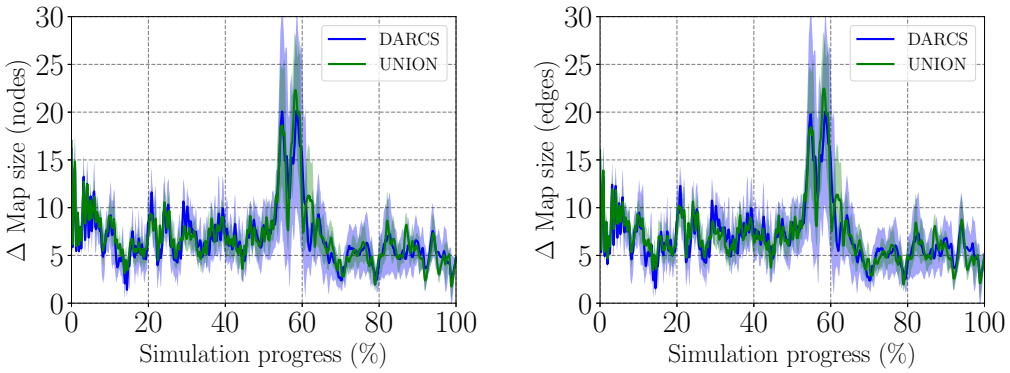
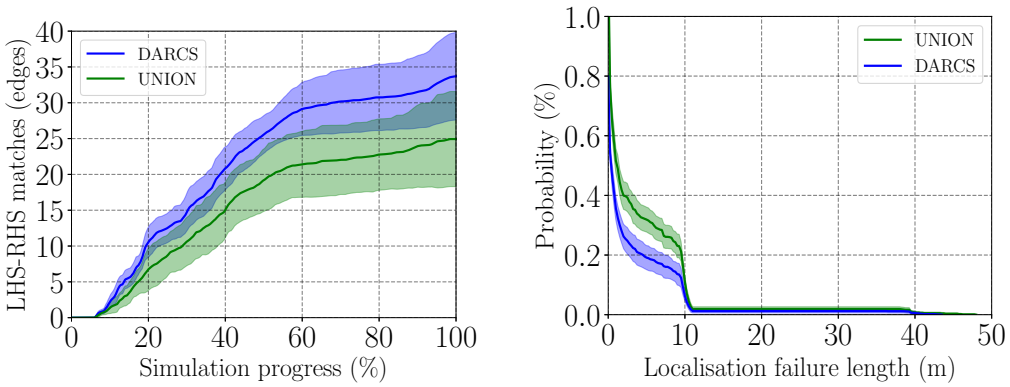


Figure 4.11: Incremental database growth under the action of distinct strategies for commuting divergent content where, even though it performs destructive operations on the *graph representation*, the Darcs *commutation policy* tracks an ideal versioning strategy in terms of (a) *node representation*, where the time series accumulates to 1181 ± 124 as compared with 1196 ± 83 nodes, and (b) *edge representation*, where the time series accumulates to 1201 ± 125 as compared with 1208 ± 83 edges. In both cases, mean disk loads are slightly lower for the Darcs *commutation policy* by the induced effect of forgetfulness through the *choice tool*. Standard deviation intervals are shown shaded (c.f. Section 2.3.2).



(a) (b)

Figure 4.12: A selective consolidation of each agent's input using Γ_{darcs} results in (a) a greater flow of related content through the DM and (b) improved localisation performance when deleting views from experiences and causing the EBN mapping module to record more information. Standard deviation intervals are shown shaded (c.f. Section 2.3.2). In (b), agents will on average experience a localisation drop-out of longer than 5 m (as an example of an unsafe dead-reckoning distance) with probability reduced from 31.89 % ± 4.16 % (UNION, green) to 19.13 % ± 4.03 % (DARCS, blue).

tion 2.4) when compared to its Union counterpart when considering incremental additions made to the *graph representation* in terms of nodes and edges entered into the working copy (Figure 4.11(a) and Figure 4.11(b)). This is attributed to the cyclical relationship between the EBN localisation and mapping modules (c.f. "plastic mapping", Section 4.2) which, when revisiting an area that has already been mapped and after having deleted some of the database content which was recorded by the navigation pipeline to be representative of that foray, may record valuable information that would have otherwise ignored (i.e. if localised, but about to lose pose). Chapter 5 presents strategies for managing this disk concern when scaling these experiments to larger MAS.

Figure 4.12(a), however, indicates that, on average, a greater number of matches are discovered during data exchanges when using Γ_{darcs} (34 ± 6 as compared with 25 ± 7 nodes as accumulated throughout each trial). Simultaneously, Figure 4.12(b) shows the associated improvement in localisation outlook, where the cumulative distributions can be summarised by marginal AUC improvements of 3.46 ± 1.41 to 3.3 ± 1.29 and a PNL improvement of $49.65\,\% \pm 15.26\,\%$ to $44.17\,\% \pm 12.14\,\%$ (c.f. Section 2.3.1). These effects are attributed to the EBN mapping module being required to encode more information into the *graph representation* to counteract the view-deletion nature of this *commutation policy*, ensuring that views which would have otherwise been discarded by an agent are presented through the DM to its peers.

4.6 Summary

With this chapter the thesis has arrived at a DM client which is compliant to a VCS interpretation of lifelong navigation.

Before the scope of the MAS extends – through the middleware tools of Section 5.3.3 – to an *ad hoc* network of communicating agents, Section 4.4.3 explored the consequences of conflating the structures of the working copy (c.f. Section 3.2) and that of graph-based SLAM techniques through a set of simulation tools in which agents act as asynchronous agents on the same filesystem.

Section 4.5.1 provided experimental verification of the convergence properties of the system with wholesale applications of the *commutation policy*, *patching tool*, and *rollback tool* to the branching model manipulations of Sec-

tion 3.3 in which no heed paid was to resource limitations. This unrestricted flow of sensory data was characterised in Section 4.3.1 and fitted through the elements of visual geometry discussed in Section 4.1 to the *node representation* and *edge representation* of Section 3.2.

Having thus encoded the spatial connectivity of the world as well as characteristics of the world at distinct places, Section 4.2 motivated the coupling of the *commutation policy* to the EBN localisation module by describing its operation in terms of metrics applied along edges in a graph structure which is also present in the *patch representation*, supported by the *search tool* described in Section 4.3.1. Section 4.5.4 thus showed how a tight relationship between lifelong navigation and this thesis' VCS interpretation acts as an effective view-deletion technique without sacrificing localisation outlook.

As a consequential implementation detail, the *choice tool* emerged in Section 4.4.2 and its implications for VCS completeness were analysed in Section 4.4.3 via the imposition of a minimum number of templated test cases which are required to exhaustively validate a rulebook for graph health as required by this chapter's candidate *commutation policy*. Section 4.5.3 showed how application of the *rollback tool* (c.f. Section 3.3.2) mitigates the associated corruption to the *graph representation*. A distributed MAS implementation of the *choice tool* itself requires communication in Section 5.3.3 of *node representation* and *edge representation* sensor data beyond the *patch representation* of changes to the local graph neighbourhood which DM agents opportunistically sample in Section 6.2.1 and naturally extend to tracking the relevance of their peer's experience to their own.

Delimiting the action of the *commutation policy* by the *product pol-*

icy (c.f. Section 3.4.1) was suggested in Figure 4.7 as an aggravating factor on the convergence outlook for the system as a whole when considering the *Patch Theory* requirements made in Section 3.1. Additionally, the prioritising of experiences for localisation as required when considering CPU performance (c.f. Section 2.4) was discussed in Sections 4.2 and 4.3.1 and is leveraged in Section 6.1.2 for correlating downloads in the centralised trading application of Chapter 5 and in Section 6.1.3 when *throttling* and *ranking* data exchanges in the distributed application. Section 6.1.4 thus suggests a targeted shopping list which agents can use to prioritise purchases (c.f. Section 3.4.2) to related parts of the *product* space.

Chapter 5

Centralised and Distributed Visual Navigation

This chapter investigates how the *canonical DM client*, instantiated in Chapter 4 and checked for conformance to the VCS interpretation of lifelong navigation of Chapter 3, behaves in a community of communicating mapping and localisation processes. Here, three realistic communication schemes are designed to support the coordination of tasks which apply the branching model of Section 3.3 in ways that both improve the average localisation outlook for the team and lend themselves well to optimisation which is strictly outside of the VCS interpretation such as the stochastic scheduling of data and provider selection in Sections 6.1 and 6.2, respectively. The consequential network, disk, and CPU burden (c.f. Section 2.4) are shown to scale well with the size of the *fleet*, where the two-agent experiments of Section 4.5 are extended to teams of arbitrary size by the introduction of middleware techniques for synchronisation of these multi-session mapping activities and

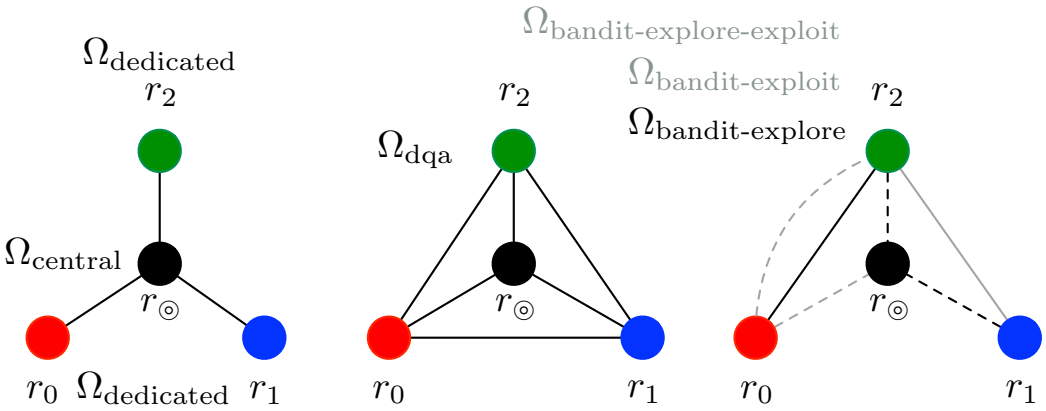


Figure 5.1: DM agents may engage in unidirectional monologues (dashed) or bidirectional dialogues (solid). A dedicated compute machine, or *central authority*, may be diligently targeted for exchanges (left, $r_\circledcirc$). In contrast, no selection of trading partner results in a *distributed-query-all* trading configuration (middle). Finally, the selection of trading partner is allowed to change between exchanges (grey, right, where r_0 and r_2 initially dialogue, but then r_2 dialogues with r_1 and r_0 monologues with $r_\circledcirc$). The random *sales policy* used in this chapter is shown (right) black. Variants of this policy are described in Chapter 6 and shown grey.

the effective delivery of *node representation* and *edge representation* data packets in isolation from the lightweight *patch representation* required for versioning.

5.1 Trading Configurations

Figure 5.1 depicts several layouts for networks of communicating DM agents, at the point of sending a request (c.f. "request" workflow, Figure 1.2).

Definition 5.1.1. *The fleet, $\mathfrak{R}$, is a collection of heterogeneously configured and equipped mobile robots, which are each identified as $r_i \in \mathfrak{R}$.*

Referring in a consistent fashion to the *team* or *fleet* by Definition 5.1.1,

we can consider the agents involved in a pairwise exchange of *patch representations* (c.f. Section 3.3.2) through the lens of a *sales policy*, Ω.

Policy 5.1.1. *At simulation or session time step k, a requesting agent, r_i, leverages a time-varying sales policy, Ω, to decide which of the fleet members it will request an exchange with as*

$$\mathfrak{R}_{\text{SELLERS}} = \Omega(\mathfrak{R},\ k),\ j \neq i\ \forall\ r_j \in \mathfrak{R}_{\text{SELLERS}} \tag{5.1}$$

such that only a single team member may be targeted (i.e. $|\mathfrak{R}_{\text{SELLERS}}| = 1$), multiple team members (i.e. $|\mathfrak{R}_{\text{SELLERS}}| \geq 1$), or none at all (i.e. $\mathfrak{R}_{\text{SELLERS}} = \emptyset$).

Here, with reference to the "respond" workflow step of Figure 1.2, agents identified by the *sales policy* consistently respond affirmatively (no rejections) and parallelise messaging on private channels assigned to multiple concurrent requesters, as will be described in Section 5.3.3. This early introduction of "sales" related nomenclature supports the targeting of DM agents which typically provide content relevant to a specific member in Section 6.2.

5.1.1 Centralised

Section 5.2.2 discusses a coordination of navigation tasks which is most appropriate for offline (e.g. overnight) learning amongst dissimilarly equipped agents with wired connectivity, where it will prove beneficial in Section 5.4.2 to optimise the order in which experiences are consolidated such that the final *graph representations* are the best that are achievable.

Definition 5.1.2. *The central authority is a robot which is privy to the sensory inputs of all agents but does not perform any mapping activities itself and is denoted as* $r_{\circledcirc}$.

Here, the *central authority* employs a *sales policy*, $\mathfrak{R}_{\text{SELLERS}} = \Omega_{\text{central}}(\mathfrak{R},\ k) = \{r_j \in \mathfrak{R},\ j \neq \circledcirc\}$, which identifies every other member of the team for exchange, who in turn do not identify any other member of the team as a source of foreign database content, $\mathfrak{R}_{\text{SELLERS}} = \Omega_{\text{dedicated}}(\mathfrak{R},\ k) = r_{\circledcirc}$. Figure 5.1 left shows the corresponding communication network, where the resemblance of the system to centralised text-based VCS (e.g. SVN [Nagel, 2005]) is clear.

5.1.2 Decentralised

In contrast, Section 5.2.3 discusses a coordination of navigation tasks which is appropriate for realtime exchange of experiences amongst similarly equipped agents with wireless connectivity.

A *distributed-query-all* approach In ideal circumstances there would be no bandwidth constraints to prevent the requesting agent from selecting for exchange partners the entire *fleet*, $\mathfrak{R}_{\text{SELLERS}} = \Omega_{\text{dqa}}(\mathfrak{R},\ k) = \{r_j \in \mathfrak{R},\ j \neq i\}$. Figure 5.1 middle depicts the corresponding communication network, where Section 5.4.6 shows experimentally how this approach aggravates the network and disk implications of Section 2.4.

A *distributed-query-one* approach To this end, before intelligent mechanisms for the selection of trading partner are presented Section 6.2, this

chapter narrows the set of peers an agent sends requests to by choosing one randomly, i.e. $\mathfrak{R}_{\text{SELLERS}} = \Omega_{\text{bandit-explore}}(\mathfrak{R},\ k) = r_j \in \mathfrak{R},\ j \neq i$.

Figure 5.1 right exemplifies the corresponding communication topology, where Section 5.4.5 shows that this approach and the centralised strategy (c.f. Section 5.1.1) scale similarly with *fleet* size.

The section which follows describes the coordination of navigation tasks around these network topologies and as such scenarios which are interpreted in Chapter 6 as simple auctions, while Section 5.3 concludes the chapter by presenting the middleware extensions which are required to support the communication of the versioning representations of Section 3.2 and the corresponding EBN packet formats of Section 4.3.1 over public and private channels corresponding to these intermittent connection opportunities.

5.2 Task Coordination

This section discusses an initialisation procedure for data exchange events (c.f. Figure 5.1) as well how sequences of such monologues or dialogues are designed around constraints on the physical system(s) inherent to the MAS and ultimately the (anticipated or measured) effect of each event upon the *graph representation*, resulting in multi-session mapping and collaborative localisation scenarios which are amenable to the bandit games of Chapter 6.

5.2.1 Processing Blocks

Figure 5.2 shows divergent *patch representations* as calculated by two agents use of the *diff tool* (c.f. Section 3.3.2).

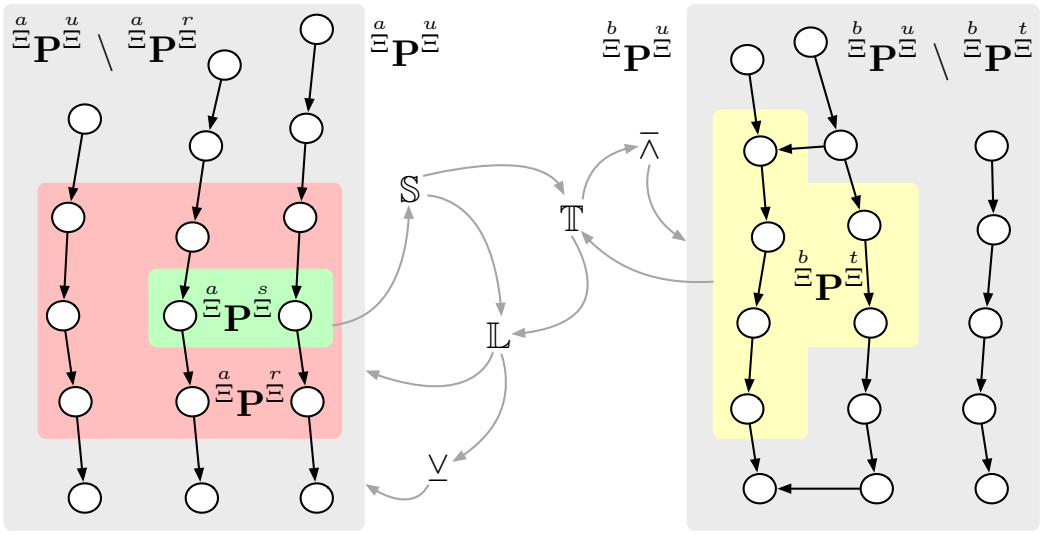


Figure 5.2: Generalised ensemble of processing blocks which are used to coordinate navigation tasks, as instances of the localisation calls described for the EBN DM client in Figure 4.2 on page 92. Requesting agents (LHS) sample some portion of the total divergence in representation ($\mathbb{S}$). Responding agents use the *node representation* data packets associated with this sample (communicated as per Section 5.3.3 below) to throttle ($\mathbb{T}$) the associated (RHS) *patch representation*. The sample finds further use in ranking ($\mathbb{L}$) the insular *graph representation* content which is likely to match to the throttled patch. Requesters and responders can optionally discard from their *history representations* the *patch representation* content not identified by the throttling and ranking procedures ($\overline{\wedge}$ and $\underline{\vee}$, respectively).

Sampling Prior to an exhaustive search for pairwise matches between the LHS and RHS patches ($^{a}_{\Xi}\mathbf{P}^{u}_{\Xi}$ and $^{b}_{\Xi}\mathbf{P}^{u}_{\Xi}$) by the *commutation policy* (c.f. Section 4.4.2), the requesting agent (left) communicates a sample of its insular *node representations* (bottom) to the responding agent (right).

Tool 5.2.1. *Requesting DM agents send a down-sampled patch representation of newly recorded content to every candidate responding agent*

$$^{a}_{\Xi}\mathbf{P}^{s}_{\Xi} = \{p \in {}^{a}_{\Xi}\mathbf{P}^{u}_{\Xi} : \mathbb{S}(p) = 1\},\ |{}^{a}_{\Xi}\mathbf{P}^{s}_{\Xi}| = \mathfrak{S} \ll |{}^{a}_{\Xi}\mathbf{P}^{u}_{\Xi}| \tag{5.2}$$

where the indicator, $\mathbb{S}$*, is referred to as the sampling tool and the threshold,* $\mathfrak{S}$*, ensures only a small subset of the divergent patch is communicated.*

In either the centralised or decentralised coordination of tasks which follows, this *sampling tool* is a nod to bandwidth limitations (c.f. Examples 2.4.4 and 2.4.5), adhering strictly to limits imposed by the medium over which samples are sent. Figure 5.2 shows an example of a down-sampled patch in green (with $\mathfrak{S} = 2$).

Throttling Focussing further on disk and network resource constraints, when requesting and responding agents are performing a data exchange based on this sample, the responding agent filters its divergent *patch representation.*

Tool 5.2.2. *The throttling tool,* $\mathbb{T}$*, leverages the value of insular content in a responding agent's graph representation to select the content*

$$ {}^{\overset{b}{\Xi}}\mathbf{P}^{\overset{t}{\Xi}} = \{p \in {}^{\overset{b}{\Xi}}\mathbf{P}^{\overset{u}{\Xi}} : \mathbb{T}(p, {}^{\overset{a}{\Xi}}\mathbf{P}^{\overset{s}{\Xi}}) = 1\}, \ |{}^{\overset{b}{\Xi}}\mathbf{P}^{\overset{t}{\Xi}}| = \mathfrak{T} \ll |{}^{\overset{b}{\Xi}}\mathbf{P}^{\overset{u}{\Xi}}| \tag{5.3} $$

which is most likely to match in a pairwise exchange, and responds to requests by sending the node representations for a maximum number of frames, $\mathfrak{T}$*, only.*

A well-implemented *throttling tool* is characterised by an intelligent selection of a request-specific subset of the insular *graph representation*, without the exclusion of information which is essential to robust localisation. Section 6.1.2 thus presents a throttling procedure which correlates the "Path

Memory" statistics (c.f. Section 4.2) accumulated in a privileged *graph representation* constructed by the *central authority* as will be discussed in Section 5.2.2 below. Figure 5.2 shows (for $\mathfrak{T} = 6$) an example of a throttled RHS patch in yellow.

Ranking Regardless of this throttle, however, agents may eventually accumulate more experiences than the CPU is able to effectively parallelise for the EBN localisation module (c.f. Example 2.4.3). Thus, after the responder has applied the *throttling tool*, a similar ranking efficiency is applied to the requester's divergent *patch representation*.

Tool 5.2.3. *The ranking tool,* $\mathbb{L}$*, leverages the value of insular content in a requesting agent's patch representation to select the content*

$$\overset{a}{\Xi}\mathbf{P}^{\overset{r}{\Xi}} = \{p \in \overset{a}{\Xi}\mathbf{P}^{\overset{u}{\Xi}} : \mathbb{L}(p, \overset{b}{\Xi}\mathbf{P}^{\overset{t}{\Xi}}) = 1\}, \ |\overset{a}{\Xi}\mathbf{P}^{\overset{r}{\Xi}}| = \mathfrak{L} \ll |\overset{a}{\Xi}\mathbf{P}^{\overset{u}{\Xi}}| \tag{5.4}$$

which is most likely to match in a pairwise exchange, up to a maximum of $\mathfrak{L}$ *frames for comparison.*

This allows the requesting agent to focus the attention of the *commutation policy* to frames that will most likely register to the sample, and are thus most likely to be registered well with the incoming content from the responder. Figure 5.2 shows an example of a ranked LHS patch in red (with $\mathfrak{L} = 9$).

Uploading To encourage flexibility in versioning, responding agents may optionally discard the throttled *patch representation* after it (and the asso-

ciated *node representation* and *edge representation* data packets) have been delivered to the requesting agent.

Policy 5.2.1. *Post-exchange versioning of the responding agent's repository representation induces, through the patching tool (c.f. Tool 3.3.2), a destructive patch representation*

$$\overset{t}{\Xi}\mathbf{P}^{\overset{b}{\Xi}} = \{(\neg\iota,\ g)\ \forall\ p = (\iota,\ g)\ \in\ \overset{b}{\Xi}\mathbf{P}^{\overset{t}{\Xi}} :\ \ \overline{\wedge}\,(p) = 1\} \tag{5.5}$$

which is appended to the history representation, and where $\overline{\wedge}(p) \in \{0, 1\}$ *is referred to as the upload policy.*

This procedure can be used to free disk resources and bears some resemblance to the "rollback" branching model manipulation of Equation 3.20 on page 71. Figure 5.2 shows (gray, right) the residual *patch representation* after such a destructive operation ($\overset{b}{\Xi}\mathbf{P}^{\overset{u}{\Xi}} \setminus \overset{b}{\Xi}\mathbf{P}^{\overset{t}{\Xi}}$).

Downloading Similarly, requesting agents may optionally discard portions of their divergent *patch representations* prior to reception of the throttled RHS patch.

Policy 5.2.2. *Pre-exchange versioning of the requesting agent's repository representation induces, through the patching tool (c.f. Tool 3.3.2), a destructive patch representation*

$$\overset{u}{\Xi}\mathbf{P}^{\overset{r}{\Xi}} = \{(\neg\iota,\ g)\ \forall\ p = (\iota,\ g)\ \in\ \overset{a}{\Xi}\mathbf{P}^{\overset{u}{\Xi}}\ \cap\ p\ \notin\ \overset{a}{\Xi}\mathbf{P}^{\overset{r}{\Xi}} :\ \ \veebar\,(p) = 1\} \tag{5.6}$$

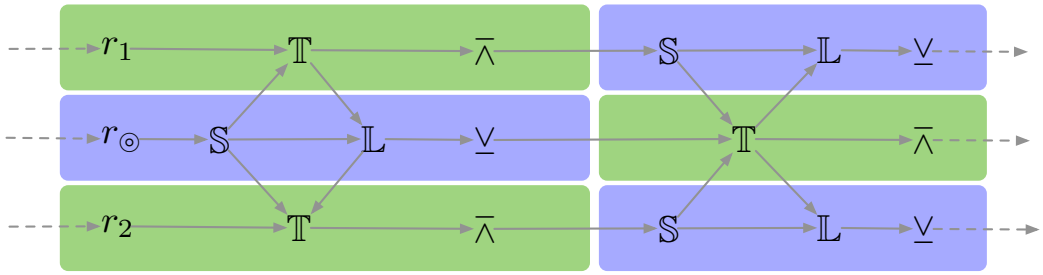


Figure 5.3: Splitting the processing of many logs amongst multiple agents is shown left where the *central authority* acts as requester (blue, $\mathbb{S} \to \mathbb{L} \to \underline{\vee}$) and subordinates act as responders (green, $\mathbb{T} \to \overline{\wedge}$). Sharing this consolidated information amongst subordinates between forays into the world is shown right, where the request/respond roles are now reversed. In the centralised trading configuration, these processing blocks ($\mathbb{S}$, $\mathbb{T}$, $\mathbb{L}$, $\overline{\wedge}$, and $\underline{\vee}$) are implemented with relaxed resource constraints for the *central authority* as compared to the subordinates, whereas the decentralised case assumes that all agents in the MAS are similarly equipped and these modules are tuned identically.

which is appended to the history representation, and where $\underline{\vee}(p) \in \{0, 1\}$ *is referred to as the download policy.*

This procedure can be used to retain only mission-critical database content, where Figure 5.2 shows (gray, left) the *patch representation* $({}^{\overset{a}{\Xi}}\mathbf{P}^{\overset{u}{\Xi}} \setminus {}^{\overset{a}{\Xi}}\mathbf{P}^{\overset{r}{\Xi}})$ of this destructive operation.

The arrangement of all of these tools and policies is dictated by the nature of the task which the *fleet* is designed to perform. This section thus concludes by describing two such ensembles for the centralised and decentralised trading configurations of Sections 5.1.1 and 5.1.2, respectively.

5.2.2 The Central Authority

Figure 5.3 illustrates a snapshot of a succession of data exchange events for the centralised trading configuration of Figure 5.1 (left). Emerging from the

depicted interaction of the processing blocks discussed in Section 5.2.1 is a pair of strategies for the management and integration of sensory data submitted asynchronously by multiple agents, which represent (a) distributed, asynchronous EBN instances on a powerful set of hardware, having split all available logs into smaller groups, or (b) several robots submitting to the *central authority* and being dealt with a subset of its privileged *graph representation* most relevant to the next foray while embarking upon regular forays into the world.

The privileged *graph representation* Figure 5.3 indicates the incorporation of two agent's maps into a centralised repository, and the selective deployment of a single map's contents, which are analogous to submitting a pull request and checking out a revision of code. The *commutation policy* acts as a view-deletion mechanism for frames with closely matching appearance (c.f. Figure 3.6 on page 67, left). Thus, as the privileged *graph representation* ($\mathcal{G}_{\circledcirc}$, as the LHS *graph representation* input to Algorithm 2 on page 105 for the *commutation policy*) is expanding to become an ever more comprehensive representation of the world, it should on average hold a greater number of *node representations* which find pairwise matches (c.f. Equation 4.18 on page 104) to incoming patches. Hence, we expect and show in Section 5.4.1 to ignore redundant information from each EBN instance, with a greater number of *patch elements* associated with *node representations* which are visually similar and passing the "Good localisation" line 15 of Algorithm 2.

Checking in Batch processing of data in this way is advantageous in that all instances of the *canonical DM client* are independent, and completed in parallel. Additionally, each instance takes as input a decimated framerate impression of live camera imagery in the form of information stored on the *node representations*, and as such the ensemble is shown in Section 5.4.1 to build maps much more rapidly than if we were to combine all logs sequentially, as in single-agent EBN, with no loss in map quality

Combinatorial space of experiences In Section 5.4.2 this rapid completion of various combinations for checking in expedites a search over some portion of the full combinatorial space of the order in which logs are collected, first attempted more greedily in [Churchill and Newman, 2013], where optimality is approximated by initially capturing more surprising and diverse information in the privileged *graph representation* as held by the *central authority*.

Checking out Figure 5.3 extends this ensemble, by considering the *team* to be periodically docking to the *central authority* in-between missions. The VCS notation of Section 3.3.2 frames this transition as a shallow copy (or clone) of the privileged *repository representation*

$$ {}^{*}_{\Xi}\mathcal{H}^{b}_{\Xi} = \{ {}^{*}_{\Xi}\mathbf{P}^{a}_{\Xi},\ {}^{a}_{\Xi}\mathbf{P}^{*}_{\Xi},\ {}^{*}_{\Xi}\mathbf{P}^{b}_{\Xi} \} \tag{5.7} $$

Section 5.4.3 shows how sharing expertise just prior to journeying out into the world again improves localisation performance at each data exchange event of this kind.

Whole-database deployment ("copy") However, $\mathcal{G}_{\circledcirc}$ may grow briskly in size and surpass the storage capabilities of a resource-constrained robot in the field. [1] Section 6.1.2 mitigates the consequential storage burden for resource-constrained *team* members by intelligently choosing which parts of the privileged *graph representation* to dispense.

5.2.3 Self-Organisation

Figure 5.3 may equivalently be considered a special instance of the decentralised trading configuration (c.f. Figure 5.1 middle and right), in which every agent has (at these instants) chosen the same team member for sending requests to (c.f. *sales policy*, Policy 5.1.1). This section thus concludes with some commentary on the implementation of the processing blocks discussed above for the centralised and decentralised trading configurations, and how this differs between agents with assigned roles (if roles are distinguishable).

Subordinate data harvesters In the centralised trading experiments presented in Sections 5.4.1 to 5.4.3, one agent ($r_{\circledcirc}$) is assumed to be equipped with more powerful hardware than the contributing subordinates. Thus the *throttling tool* and *upload policy* as implemented by subordinates faithfully copy each and every node to the privileged *graph representation* with no exclusions and discards all checked in content. In contrast, the *ranking*

[1] Consider that Wi-Fi speeds are typically 162.5 MB s^{-1} and that a 16 bit 384 px×512 px colour image is 0.375 MB, allowing for 433 images to be communicated per second. This deployment scheme may therefore not suitable for realtime use, as it would require 22 000 images to be communicated (i.e. for fifty-five dataset logs each traversing approximately 800 m and representing 1600 image *node representations* spaced out in 0.5 m increments), which would take approximately 51 s to deploy.

tool as implemented by the *central authority* ensures that incoming frames are checked for correspondence to the entire *graph representation*, which is not cleared of potential matching frames by the *download policy*.

Collaborating localisers In the decentralised trading configuration, resources are assumed to be uniform and all processing blocks are implemented identically for every DM agent as shown in Figure 5.2, with the common goal of limiting the number of parallel processes deployed by the *canonical DM client* to maintain constant-time performance. To investigate the worst-case burden on these collaborator's resources, the experiments presented in Sections 5.4.4 to 5.4.6 employ a *upload policy* and *download policy* which do not clear the responding or requesting agents *graph representations* of communicated *patch representations*.

5.3 Networking Protocol Suite

This section introduces additional representations and tools that DM agents use to interface with one another, and designers can use to explore the combinatorial space of policies presented throughout this thesis. Specifically, we discuss a network layer which is used to synchronise agents at communication opportunities (such as Figure 5.3) and allows *fleet* operations to be hosted on multiple computers and not on the same filesystem, departing from the *skeleton tool* (c.f. Section 4.4.3) simulated versioning experiments of Section 4.5. The chapter concludes by detailing the separate communication of *patch representation* and the associated *node representation* and *edge*

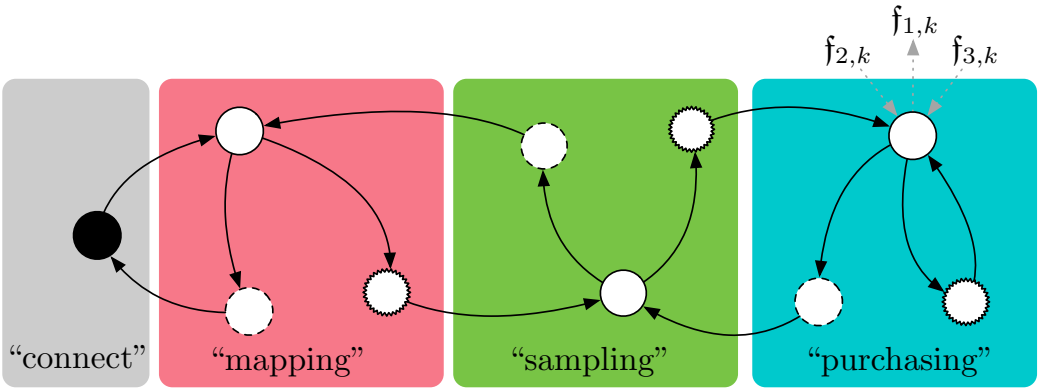


Figure 5.4: Visualisation of the state transition table for a single robot entering in the DM (grey), and then iteratively mapping (red), sampling (green), and purchasing (blue). Each transition is read as one of "waiting" (circles), "executing" (jagged circles), or "leaving" (dashed circles).

representation data packets during execution of data exchanges as required to implement the processing blocks of Section 5.2.1.

5.3.1 A Multirobot Finite-State Machine

Each DM agent is considered as an abstract finite-state machine (FSM), depicted in Figure 5.4, which can be in exactly one of a finite number of states at any point in time, and can respond to internal or external stimuli [Bosik and Uyar, 1991].

Representation 5.3.1. *At simulation time steps, k, each DM agent, r_i, publishes a flow representation, $\mathfrak{f}_{i,k} = (\theta_{i,k}, \psi_{i,k})$ where $\theta_{i,k}$ is a state descriptor (e.g. "mapping", "sampling", "purchasing") and $\psi_{i,k}$ is a semaphore primitive [Kosaraju, 1973] which counts iterations of each state-type and provides entry and exit synchronisation points for agents which may employ wildly distinct policies.*

In the single-agent case, the FSM only responds to internal stimuli. In the centralised case, the FSM considers as external stimulus only the *central authority*. In the fully distributed case, however, the FSM can transition from one state to another in response to encoded stimuli from all, or some, other members of the *fleet*.

5.3.2 Synchronisation Interfaces

In the interests of facilitating this FSM, this thesis presents a middleware extension to the Mission-oriented Operating Suite (MOOS) [Newman, 2008], with a public interface that guarantees synchronisation for *fleet*s of robots which are alternatively recording and sharing mapping data.

Example 5.3.1. *Figure 5.4 shows the behaviour of the FSM for a DM agent, r_0, which does not transition from "waiting to purchase" (blue, circle) to "purchasing" (blue, jagged circle) until the corresponding flow representation has been communicated over a public channel by every other agent (r_1 and r_2) in the MAS.*

Figure 5.4 shows the inter-relationship between the instances of the *flow representation*, where the initial (and terminal) state shown black correspond to an agent leaving the network community and is equivalent to the "start" of workflow of Figure 1.2 on page 4.

The MOOS network community is typically hosted in the cloud[2], but may be served up by one DM agent (e.g. the *central authority*). Section 5.3.3 describes a mixed use of public and private MOOS channels for executing a

[2]Cloud infrastructure such as `https://azure.microsoft.com/en-gb/services/virtual-machines`

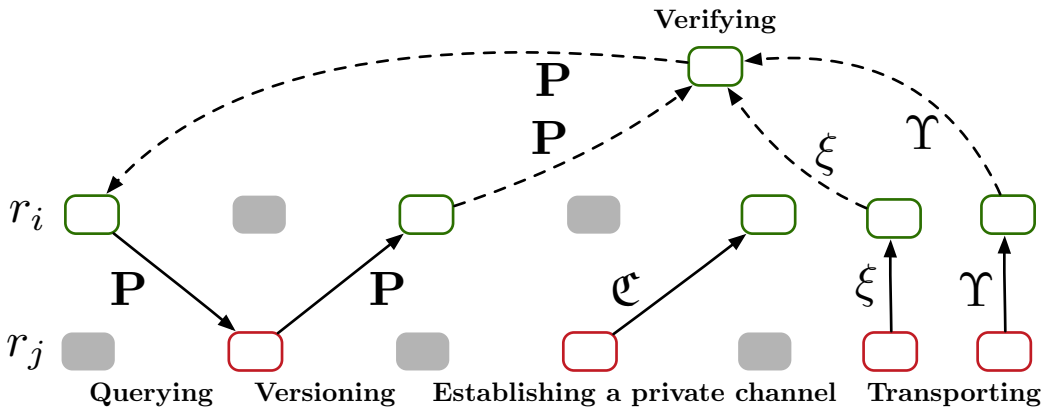


Figure 5.5: Iteratively communication of lightweight patch messages and the establishment of a private channel over which the corresponding *edge representations* and *node representations* are communicated, which are then verified. Communicated messages between the responding agent (r_j) and the requesting agent (r_i) are solid black, while data structures for multi-threading the exchange on the requesting agent's CPU are shown dashed. Operations performed by the responder and requester are indicated as red and green, respectively, while instants at which either agent waits on a pending (internal, or external) operation are shown grey.

data exchange, whereas the *flow representation* utilises strictly public channels which are preset before mapping and trading can commence.

5.3.3 Executing a Data Exchange

Figure 5.5 illustrates the Transmission Carrier Protocol (TCP) threeway handshaking procedure [Comer, 1988] that DM agents executing an exchange follow, after establishing a communication opportunity (c.f. Figure 5.1) and synchronising *flow representation* as discussed in Sections 5.3.1 and 5.3.2.

The operations discussed thus far in this thesis which are affected upon and between *graph representations* which require *node representation* and *edge representation* data packets are executed in the following step-wise fash-

ion:

1. **Querying**: The requesting agent sends a *patch representation* of its *graph representation* (limited to the *product* of interest by the *category tool*, c.f. Tool 3.4.2) to the responding agent over a public channel which is listened to passively.

2. **Versioning**: The responding agent uses the *diff tool* (c.f. Section 3.3) to send a smaller *patch representation* back to the requesting agent.

3. **Establishing a private channel**: The responding agent creates (again using UUIDs, c.f. Section 4.1.2) a unique channel identifier, $\mathfrak{C}$, and shares it with the requesting agent.

4. **Transporting**: The responding agent opens the private channel and publishes the *edge representation* and *node representation* data packets corresponding to the *patch representation* of item 2.

5. **Verifying**: The requesting agent opens the private channel, waits for it to go silent, and then compares the reconstructed *patch representation* with what it was expecting.

Packet losses If the patch ultimately received is incorrect or incomplete (c.f. item 5), the requesting agent has the choice of accepting a configurable number of packet losses, or repeating the above procedure until all losses have been mitigated, on each iteration needing to initially request a much sparser patch than before. Most TCP transfers are short enough to see a multitude

of losses and consequently their performance is dominated by startup effects such as connection establishment and slow start [Cardwell et al., 2000].

The steps described by item 1 to item 5 above are relevant to all model manipulations which do not consider in isolation the UUIDs (c.f. Section 4.1.2) native to the *patch representation* used for versioning (e.g. Equations 5.2 to 5.6, line 13 of Algorithm 2, and the *choice tool* – Tool 4.4.1), where it is implied that this handshaking procedure is inherent to those implementation details.

5.4 Results

As a corollary of the instantiation in Chapter 4 of the *canonical DM client* through the VCS notation of Chapter 3, this section tests the impact upon localisation outlook for MAS which are collaborating over a physical network via the communication topologies discussed in Section 5.1 with no shared filesystem.

In particular, we are concerned with the consequential trade-off against the amount of CPU, disk, and network resources dedicated to ideal exchanges of *graph representation* as a result of wholesale implementations of the processing blocks discussed in Section 5.2.1, anticipating the experience-selection techniques of Section 6.1.3. These resource effects are shown to scale well with the size of the team under conditions of incomplete membership of data exchange events during coordination of navigation tasks by the *distributed-query-one* trading configuration discussed in Section 5.1.2, providing the framework for simultaneously targeting content and provider in Chapter 6.

Sections 5.4.1 to 5.4.3 presents results which were obtained by 5-fold

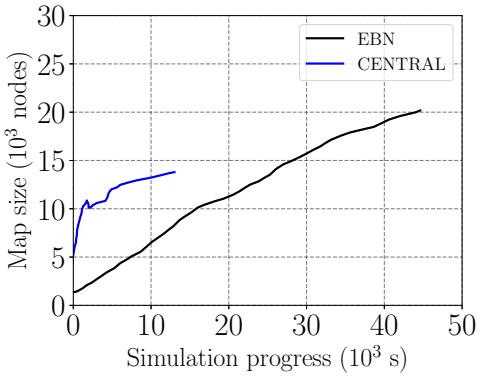


Figure 5.6: Cumulative node edge counts averaged over the team. Devolution of map building to subordinates (blue) yields the map in 13 120 s (3.6 h), while single-agent EBN (black) requires 44 690 s (12.4 h). The dip exhibited in the node count through the centralised assembly technique (CENTRAL, blue) is due to destructive pre- and post-exchange versioning operations performed by the *central authority* via the *upload policy* and *download policy* (c.f. Policies 5.2.1 and 5.2.2).

cross-validation (c.f. Section 2.3.2) over the BEGBROKE-24HR urban dataset (c.f. Appendix A.2.1, using the experience segment *product policy* of Section 3.4.1).

In contrast, Sections 5.4.4 to 5.4.6 aggregates statistics over randomised trials with no held-out sets (c.f. Section 2.3.3) using video sequences from ten of the *product* space regions specified by the GPS catalogue *product policy* over the *Oxford RobotCar Dataset* (c.f. Appendix A.2.2).

5.4.1 Building Good Maps Quickly

This experiment investigates distributing the building of an EBN database from all logs in the cross-validation set while checking in intermediate results as opposed to sequential processing of the same logs by one agent in isolation. Here, two agents perform five cross-validation trials, each of which consists of

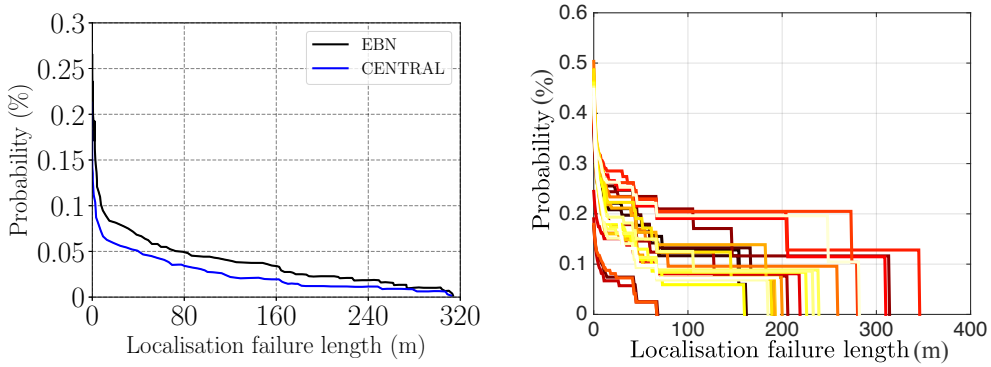
a group of nine dataset logs which traverse the 800 m extent of the ρ_{BEGBROKE} *product*, and communicate (after processing all logs in each group) via the centralised trading configuration and a third agent which acts as the *central authority*.

Figure 5.6 indicates that the privileged *graph representation* is built in a fraction of the traditional time, specifically 29.36 %. The final privileged *graph representation* node count is on average 67.57 % of that generated by single-agent EBN, reaching 13 665 nodes as compared with 20 222 nodes. The smaller database sizes are attributed to the presence of a greater number of localisers (represented by the subordinates) in agreement which disables the recording of new map content (c.f. Section 4.2).

Figure 5.7(a) shows that no significant compromise is made to map quality and localisation robustness is commensurate – where the held-out set spends 21.97 % of its foray not localised in the map built by subordinates, compared with 32.03 % in the map built with single-agent EBN. One would expect to localise at best as well as single-agent EBN, and no better, suggesting that injudicious incorporation of all sensory inputs at every opportunity propagates the effect on topometric accuracy of one bad stereo match.

5.4.2 Optimising Map Construction

Indeed, this map building strategy allows for coarse but rapid combinatorial optimisation of the order in which the privileged *graph representation* is constructed, and this experiment thus investigates the effect of the order in which dataset logs are handed to the *canonical DM client* on the quality of



(a) (b)

Figure 5.7: (a) Devolution of map building to subordinates (blue) does not compromise on map quality, offering localisation robustness that is at least comparable to single-agent EBN (black) and (b) rehearsal of the order in which video sequences are processed by subordinates allows for rapid (in 22.44 % of the time as compared to single-agent EBN) exploration of the combinatorial space of experiences. In (a), for a safe dead-reckoning driving distance of 160 m, the centralised map assembly method reduces the probability of experiencing an unsafe failure from 3.37 % (CENTRAL, blue) to 1.96 % (EBN, black) as averaged over all dataset logs within the five held-out groups.

representation. Here, two agents combine (in two check in stages) four groups of nine logs in every possible order over the 800 m extent of the ρ_{BEGBROKE} *product*, and communicate via the centralised trading configuration and a third agent which acts as the *central authority*, totalling $4! = 24$ candidate maps.

Figure 5.7(b) shows the quality of each resulting privileged *graph representation* when localising the held-out set in which it is clear (i.e. green, blue, and orange curves closest to the origin) that the designer is in possession of a best quality map for deploying[3]. This is attributed to an idealised recording

[3]The design choice depends on the distance robots can reasonably travel on dead-reckoning alone, which could be dictated by safety, quality of ego-motion estimation avail-

of the most surprising experiences in the map at the earliest opportunity.

It can be shown[4] that the (fractional) time saved for exploring the combination of k groups of stereo logs is

$$\frac{(N-1)T_G + (N-1)!(N-2)T_M + (N-1)((N-1)!-1)T_C}{((N-1)!(N-2)+1)T_G + ((N-1)!-1)T_C} \tag{5.8}$$

where in this case there are $N = 5$ cross-validation groups (c.f. Section 2.3.2). The advantage of distributing mapping efforts to asynchronous processes is clarified via Equation 5.8, as the copying of intermediate results for single-agent EBN is limited to only the first group of processed logs (subsequent processing depends on the map built using imagery from the first group). Hence, under this framework, all intermediate results can be copied, and we are rapidly searching for the best map in 22.44 % of the traditional time.

5.4.3 Centralised Sharing of Expertise

This experiment examines the utility of the privileged *graph representation* as constructed by the methods investigated in Sections 5.4.1 and 5.4.2 (arbitrary or optimal, respectively).

Here, two robots (r_1 and r_2) each embark on seven forays into the world (each

able, or frequency of localisation achievable with the available processing power.

[4] $T_G = 10\,823\,\text{s}$ is the time for processing each group, $T_M = 1829\,\text{s}$ is the time to check in one agent's *graph representation* into the *central authority*, and $T_C = 27\,\text{s}$ is the time to copy intermediate results for reuse (all measured from jobs submitted to a computing cluster).

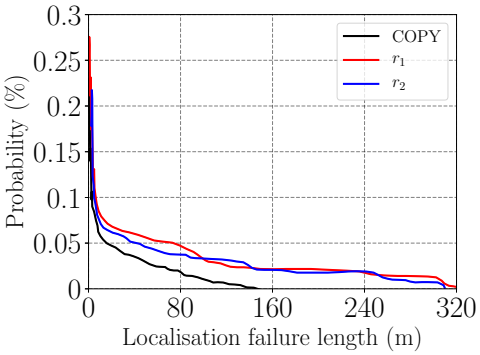
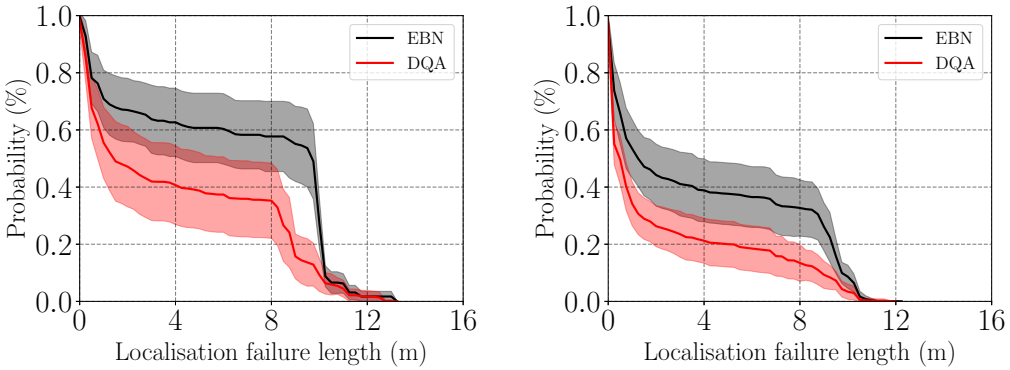


Figure 5.8: Centralised trading of navigation data (COPY, black) with cumulative failure distributions. The held-out set is able to localise more reliably using the shared map (black) than within either of the per-robot maps (red and blue for r_1 and r_2, respectively) that each robot would be limited to if lacking knowledge of each other. As the number of robots in the experiment increases, the cumulative failure distributions should hug the vertical axis more closely.

over the 800 m extent of the ρ_{BEGBROKE} *product*) on two occasions, pausing between forays to check in to the *central authority*, clear their insular *graph representations* (c.f. *upload policy*, Policy 5.2.1) and finally checking out the resulting privileged *graph representation* in its entirety (i.e. "copy", Equation 5.7).

For a complete treatment of item 4 of the "guiding questions" on page 5, Section 6.4.1 presents experiments in which data exchanges at these deployment opportunities are *throttled* by the *central authority*. At this point, however, Figure 5.8 shows the advantages offered by an unrestricted sharing scheme, in which it is strikingly evident on inspecting the distributions of localisation failure that *graph representation* quality in the case of consolidating expertise between these agents is better than if the robots had no knowledge of each other and had built *graph representations* with only their own sensory input.



(a) Cumulative BEVINGTON failures.

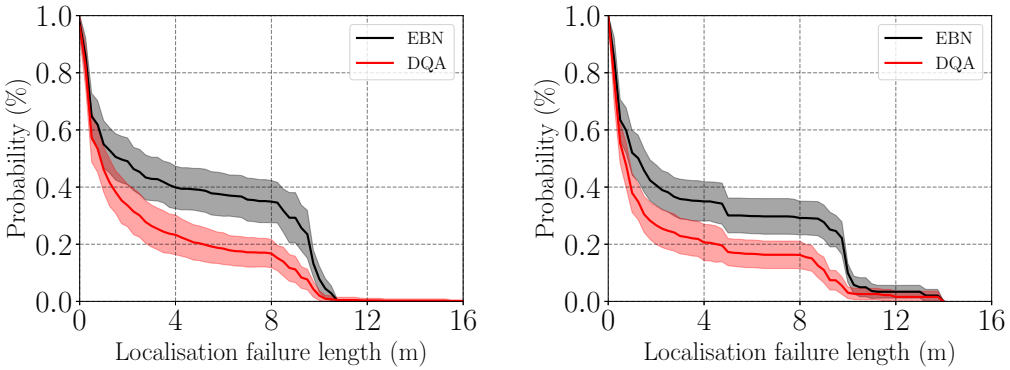
(b) Cumulative BROAD failures.

Figure 5.9: Decentralised *distributed-query-all* (DQA, red) trading of navigation data bolsters localisation robustness as compared to single-agent EBN (black). For a safe dead-reckoning driving distance of 5 m, the *distributed-query-all* technique reduces the probability of experiencing an unsafe failure from 60.71 % ± 12.14 % to 38.74 % ± 13.36 % and from 37.66 % ± 10.80 % to 20.01 % ± 7.99 % for (a) and (b) respectively.

Here, for a safe dead-reckoning driving distance of 80 m, the whole-database deployment method reduces the probability of experiencing an unsafe failure from 4.63 % (r_1, red) and 3.70 % (r_2, blue) to 1.74 % (COPY, black) as averaged over all dataset logs within the five held-out groups. Simultaneously, the AUC is improved from 10.16 (r_1, red) and 8.72 (r_2, blue) to 3.75 (COPY, black).

5.4.4 Decentralised Sharing of Expertise

To address "guiding question" item 3 on page 5, this experiment follows that of Section 5.4.3 by demoting the *central authority* such that the MAS is populated by equally-equipped collaborating localisers (c.f. Section 5.2.3). Here, we perform ten trials in which four agents engage in thirteen forays over a 70 m portion of the $\rho_{\text{BEVINGTON}}$, ρ_{BROAD}, ρ_{ORI} and ρ_{TRINITY} *products* and



(a) Cumulative ORI failures.

(b) Cumulative TRINITY failures.

Figure 5.10: Decentralised *distributed-query-all* (DQA, red) trading of navigation data bolsters localisation robustness as compared to single-agent EBN (black). For a safe dead-reckoning driving distance of 5 m, the *distributed-query-all* technique reduces the probability of experiencing an unsafe failure from 39.04 % ± 7.49 % to 20.32 % ± 6.06 % and from 30.07 % ± 6.07 % to 17.27 % ± 4.76 % for (a) and (b) respectively.

use the *sales policy* Ω_{dqa} to execute trades with every other agent at every opportunity (c.f. Section 5.1.2).

As for the centralised trading configuration (c.f. Figures 5.7(a), 5.7(b) and 5.8) it is clear from Figures 5.9 and 5.10 that unrestricted exchanges navigation data through the decentralised trading configuration is beneficial to localisation outlook, where the cumulative failure distributions are consistently improved.

Tables 5.1 and 5.2 provide related statistical indication of this idealised improvement in localisation outlook through the *distributed-query-all* approach (c.f. Figure 5.1 middle), where each entry corresponds to an experiment of ten trials in which four agents engage in thirteen forays over a 70 m portion of the $\rho_{\text{BLAVATNICK}}$, $\rho_{\text{GLOUCESTER}}$, ρ_{MUSEUM}, ρ_{PARKS}, ρ_{PHYSICS}, and ρ_{PITTRIV} *products*.

	MUSEUM	PARKS	PHYSICS	PITT-RIVERS
EBN(100 %)	3.79 ± 1.82	3.47 ± 1.09	6.17 ± 1.43	5.77 ± 1.60
DQA	3.20 ± 1.82	1.51 ± 0.60	4.46 ± 1.30	3.01 ± 0.86

Table 5.1: Improved localisation performance over two regions known to the GPS catalogue *product policy* as a result of a *distributed-query-all* (DQA) trading configuration, enjoying 84.22 %, 43.39 %, 72.30 %, and 52.19 % of the area-under-curve (AUC, c.f. Section 2.3.1) available to a single *canonical DM client*.

	BLAVATNICK	GLOUCESTER
EBN(100 %)	$59.74\,\% \pm 11.38\,\%$	$79.41\,\% \pm 13.26\,\%$
DQA	$43.53\,\% \pm 15.30\,\%$	$53.77\,\% \pm 17.78\,\%$

Table 5.2: Improved localisation performance over two regions known to the GPS catalogue *product policy* as a result of a *distributed-query-all* (DQA) trading configuration, enjoying 72.85 % and 67.71 % of the percentage-not-localised (PNL, c.f. Section 2.3.1) rate available to a single *canonical DM client*.

Note (for the *distributed-query-all* approach, red) the difference in tail lengths in the extreme cases of Figure 5.10(a), in which the most severe failure was 17.94 m, and Figure 5.9(b), where the tail reaches only 12.17 m. Chapter 6 recognises that regions of the environment present distinct levels of difficulty to the EBN system (e.g. in areas of high visual aliasing) and uses the itemised data exchange protocol discussed in Section 3.4.2 to leverage instrumentation at this spatial resolution such that *fleet* learns a nuanced model for the value that local experts provide in parts of the world.

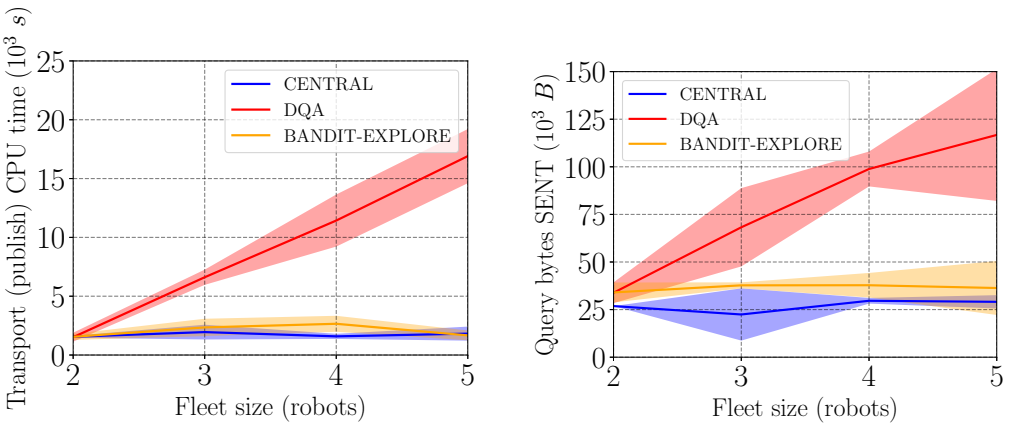


Figure 5.11: Investigating performance degradation in multi-robot *fleet*s of increasing membership showing averages for the cumulative (a) CPU time spent publishing *node representation* data packets for consumption by another agent and (b) query *patch representations* sent over the network as functions of the *team* size. Unrestrained exchanges of data with every *fleet* member (DQA, red) scales linearly with the scope of the MAS, whereas unrestrained exchanges with a dedicated *fleet* member (CENTRAL, blue) and a random selection of just one trading partner (BANDIT-EXPLORE, orange) show flat resource requirements with the size of the *team*.

5.4.5 Scaling Marketplace Interactions

Addressing "guiding question" item 2 on page 5, this experiment departs from the idealised, fully-connected collaboration of localisers in Section 5.4.4. Figure 5.11 investigates how the trading strategies discussed thus far scale with the size of the *team*, aggregating CPU and network loads over ten trials in which two to five agents engage in ten forays over a 40 m portion of the ρ_{BROAD} *product* and use the *sales policy* $\Omega_{\text{bandit-explore}}$ to execute trades with a randomly selected peer at each connection opportunity (c.f. "connect" in the workflow of Figure 1.2 on page 4).

Figure 5.11(a) shows averages (over the team and over all trials) for the cumulative time spent responding to requests by publishing *node representation* data packets over MOOS (c.f. "transporting" in Figure 5.5). The *distributed-query-one* approach stays relatively flat for teams of four agents, spending 1668.30 s ± 418.58 s as compared to 1545.44 s ± 260.14 s for a pair of agents using the same strategy. The *distributed-query-all* strategy exhibits an undesirable change in performance, moving from 1516.20 s ± 381.52 s to 16 907.48 s ± 1994.68 s over the same expansion of membership.

Figure 5.11(b) shows similar averages for the cumulative *patch representation* bytes as used by the *sampling tool*, *ranking tool*, and *throttling tool* in the interaction of the processing blocks discussed in Section 5.2.1 from the requesting agent's perspective. The *distributed-query-one sales policy* remains relatively constant for teams of four agents, publishing 36.33 kB ± 7.23 kB as compared to 34.01 kB ± 2.66 kB for a pair of agents using the same strategy. Similarly to the CPU effect observed in Figure 5.11(a), the *distributed-*

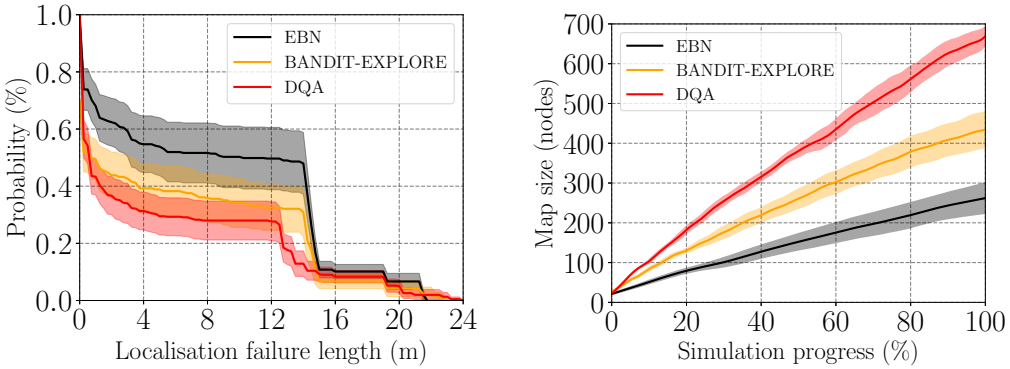


Figure 5.12: (a) Decentralised *distributed-query-one* (BANDIT-EXPLORE, orange) trading of navigation data approaches the localisation performance of the idealised *distributed-query-all* alternative (DQA, red). For a safe dead-reckoning driving distance of 10 m, the *distributed-query-one* technique reduces the probability of experiencing an unsafe failure from 50.27 % ± 17.96 % (EBN, black) to 39.00 % ± 9.72 % and approaches the idealised 27.98 % ± 11.34 %. (b) Average (over the team and trials) accumulated number of nodes in *patch representations* affected upon the *graph representation*, reaching a relaxed final node count of 434 ± 47 nodes as compared to 669 ± 26 for both trading configurations (orange and red, respectively), approaching the overall disk load for single-agent EBN of 262 ± 40 nodes.statistical effects on results over more than one trial

query-all strategy exhibits an undesirable change in performance, moving from 33.85 kB ± 2.78 kB to 116.75 kB ± 17.69 kB over the same change in team size.

By both measures, performance degrades as the size of the *fleet* increases, unless a narrow selection of trading partner is made and the scaling observed matches coordination around the *central authority*. Section 6.2 presents a guided alternative to the random selection used here.

5.4.6 Mitigating Resource Issues

This experiment exposes auxiliary aspects of the narrow selection of responding agent in Section 5.4.5 as well as the benefit of said approach to the resourcing concerns of Section 2.4 even within teams of fixed membership. Figures 5.12 and 5.13 thus aggregate instrumentation over ten trials in which three agents once again engage in five forays over a 30 m portion of the $\rho_{\text{BLACKHALL}}$ *product*. The agents communicate using either the *distributed-query-all sales policy* with no throttling or ranking restrictions (i.e. $\mathbb{T}$ and $\mathbb{L}$ are set arbitrarily high) or a random selection of trading partner through the *distributed-query-one sales policy*, with similarly random thresholds, where $\mathbb{T}$ and $\mathbb{L}$ are tuned to sample *node representations* randomly. Section 6.4.3 presents an alternative implementation of the *throttling tool* and *ranking tool* tuned for FAB-MAP compatibility within regions of the *product policy*.

Figure 5.12(a) shows a drop-off in localisation performance when agents are forced to restrict the scope of team members to send requests to (c.f. Equation 5.1 on page 128). This is attributed to a departure from exactly-

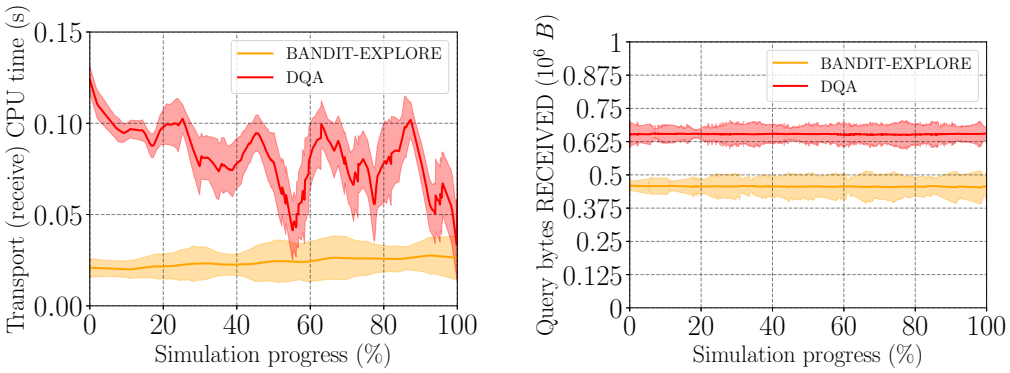


Figure 5.13: Average (over the team and trials) (a) time spent in reception of requested *node representation* data packets, accumulating to a relaxed CPU load of $0.36\,\mathrm{s} \pm 0.04\,\mathrm{s}$ (BANDIT-EXPLORE, orange) as compared to $2.35\,\mathrm{s} \pm 0.13\,\mathrm{s}$ (DQA, red) and (b) bandwidth required for reception of said packets, accumulating to a relaxed network load of $47.75\,\mathrm{MB} \pm 5.70\,\mathrm{MB}$ as compared to $306.98\,\mathrm{MB} \pm 4.53\,\mathrm{MB}$.

synchronised mutual *history representations* (c.f. Figure 3.5(b) as opposed to Figure 3.5(a)) inherent to the experiments presented in Section 4.5.

Figure 5.12(b) shows the associated accumulation in insular *node representations*, in which it is clear that this trading configuration is ultimately more lenient on the disk.

Indeed Table 5.3 shows final node counts for the narrow selection of trading partner which are much closer to conditions of no communication (EBN). Here, each tabulated result represents ten trials in which three agents engage in thirty-four and eighteen forays over 10 m portions of the $\rho_{\text{NORTH-STORE}}$ or $\rho_{\text{PHOENETICS}}$ *products*, respectively.

Conversely to Figure 5.11(a), Figure 5.13(a) shows the time series (not accumulated) CPU time spent receiving (not publishing) *node representation* data packets over the network by responding agents, which is evidently

	NORTH-STORE	PHONETICS
EBN(100 %)	354 ± 83	156 ± 21
BANDIT-EXPLORE	490 ± 76	239 ± 18
DQA	891 ± 39	398 ± 25

Table 5.3: Improved end-of-simulation node counts as a result of a *distributed-query-one* (BANDIT-EXPLORE) trading configuration, averaging 138.34 % and 153.09 % of the disk loads induced by single-agent EBN over two regions known to the GPS catalogue *product policy* as compared to 251.60 % and 255.16 % for the *distributed-query-all* strategy (DQA).

more lenient when using $\Omega_{\text{bandit-explore}}$.

Likewise, the network load shown in Figure 5.13(b) differs from Figure 5.11(b) by exposing the underlying time series as well as capturing the bandwidth dedicated to receiving (not publishing) *node representation* data packets by requesting agents. Here, bandwidth restrictions are managed by the narrow selection of trading partner, where the *distributed-query-one* strategy exhibits a negative offset from the *distributed-query-all* strategy and accumulates over the course of all trading opportunities to $47.75\,\text{MB} \pm 5.70\,\text{MB}$ as compared to $306.98\,\text{MB} \pm 4.53\,\text{MB}$.

5.5 Summary

This chapter has discussed idealised coordinations of mapping and localisation tasks for communities of agents, each of which conform – as a *canonical DM client* – to the working copy *repository representation* (c.f. Section 3.2.1) as shown in Section 4.3 where the EBN localisation module (c.f. Section 4.2) is applied exhaustively to exchanged *patch representations* as a proxy format for *graph representations* (c.f. Section 4.4).

Section 5.2.1 presented the *sampling tool*, *throttling tool*, and *ranking tool* as mechanisms for requesting agents to instigate data exchanges and for both requesting and responding agents to curtail the scope of the branching model manipulations of Section 3.3.2 based on a measured relevance to the request. These *patch representation* selectors were assembled in Section 5.2.3 as two separately motivated MAS organisations, including teams of (a) subordinate data harvesters regularly "checking in" processed video sequences to a *central authority* (c.f. Section 5.2.2), which is able to deploy a privileged *graph representation* by a "check out" procedure which obeys distinct resource limitations and (b) collaborating localisers which rely on uniform implementations of these "processing blocks" across the *fleet*.

Data exchange opportunities were described as arising from the communication topologies of Section 5.1, where the *sales policy* was introduced as the selection mechanism which requesting agents use to target a desired responding agent. The execution of these monologues or dialogues over a shared middleware network layer was described in Section 5.3.3, where the communication of the unique identifiers (c.f. Section 4.1.2) native to the *patch representation* format was distinguished from communication of the corresponding *node representation* and *edge representation* data packets required to guide the versioning operations of Chapters 3 and 4 (e.g. the *choice tool* and consequently the *commutation policy*).

Sections 5.3.1 and 5.3.2 presented further use of this network layer and matured the adjacent DM simulation framework (c.f. *skeleton tool*, Section 4.4.3) by discussing a *flow representation* and handshaking protocol (c.f. Figure 5.4) which agents use to manage the progress of their multi-

session mapping activities.

Sections 5.4.1 to 5.4.3 and Sections 5.4.4 to 5.4.6 verified experimentally the idealised boosts to localisation performance available through the centralised and decentralised trading configurations, respectively, when no restrictions are placed on the *throttling tool* or *ranking tool*. Section 6.1 describes how the details emerging from realistic implementations of these tools are used to focus data exchanges between *patch representations* which are likely to be compatible through the EBN localisation module. Indeed, while Sections 5.4.5 and 5.4.6 show that a narrow selection of trading partner induces more lenient CPU, disk, and network burdens and scales well with the size of the *fleet*, this targeting of content then allows Section 6.2 to measure and anticipate the "price" of a data exchange to guide the selection of trading partner through a discriminative *sales policy*.

Chapter 6

Trading in Lifelong, Multi-Agent Localisation

The discussion thus far has restricted the use of quality of *graph representation* to the *commutation policy*, as it employs the *choice tool*. In Chapter 5 the boon to localisation outlook for the compliant DM client of Chapter 4 when privy to all (or some random subset) of the knowledge shared in a consistent fashion via the VCS interpretation of Chapter 3 by a collaborating team or *central authority* was verified. This chapter recognises agents as simultaneously acquiring new knowledge and optimising decisions based on existing knowledge and matures the MAS task coordination of Section 5.2 by balancing (i.e. by the resourcing concerns of Section 2.4) these competing tasks in order to maximise their total value. Specifically, Section 6.1 presents an efficient coarse-to-fine transformation of the team's operational envelope (i.e. the world) which applies experience-selection strategies to the implementation of the generalised processing blocks discussed in Section 5.2.1 (in

contrast to the unrestricted peer-to-peer experiments of Section 5.4) over an itemised *product* space which focuses versioning activities (c.f. Section 3.4.2). Finally, Section 6.2 discusses a method for agents to track and exploit the time-evolving utility of a peer's experiences as a principled replacement for the random selection of trading partner in Section 5.1.2.

6.1 Targeting Content

This section discusses efficient implementations for the processing blocks of Section 5.2.1, with specific focus on the *ranking tool* and *throttling tool* – implementations for which were unrestricted in order to maximise localisation outlook in Section 5.4 – framing the centralised and decentralised coordination of visual navigation tasks in Chapter 5 as contextual relationships during scheduling of a batch of stochastic jobs in Section 6.2.

6.1.1 Checking in Asynchronously

During "check in" for the centralised coordination of tasks (c.f. Figure 5.3, left) subordinate agents act as *sellers* while the *central authority* acts as the sole *buyer*. Figure 6.1 shows an exchange of patches between agents which are assigned these roles.

Sampling segments To facilitate the statistical correlation of "checking out" activities in Section 2.3, we must consider that "Path Memory" creation (c.f. Section 4.2) is done incrementally at run-time, using likelihood terms which are interpreted as probabilities over a robot's live trajectory

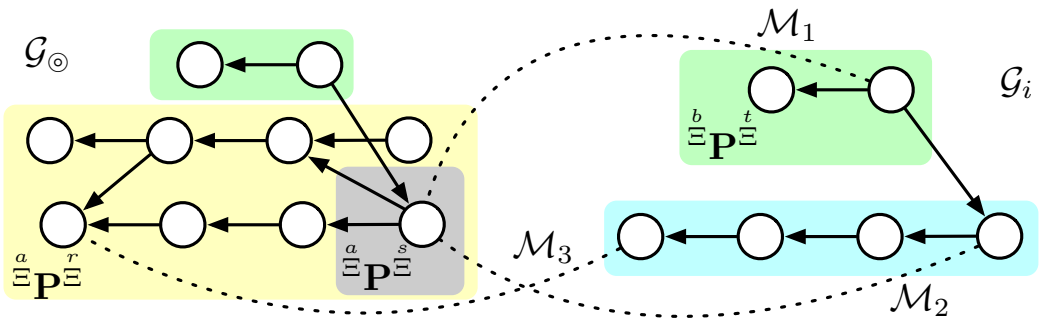


Figure 6.1: In the centralised versioning framework, robots hand off sensory information to the privileged *graph representation* ($\mathcal{G}_{\circledcirc}$, left) in the form of an independently processed *graph representations* ($\mathcal{G}_i$, right). Agents send insular database content with no throttling (green and blue), the *patch representations* for which are matched by the *central authority* with no preferential ranking applied. If these localise well (dashed lines), they are omitted (blue, $\mathcal{M}_2$ and $\mathcal{M}_3$) from the privileged map, driven by the natural behaviour of localisation as performed by EBN. If localisation is not sufficiently good (green, only $\mathcal{M}_3$) this information is regarded as required and it is included within and connected to $\mathcal{G}_{\circledcirc}$.

[Linegar et al., 2015]. Thus, in contrast to the random submissions of Sections 5.4.1 and 5.4.2, the *central authority* uses the *sampling tool* (c.f. Equation 5.2 on page 131) to request that subordinates check in database content which appends segments (going forward in time) for the centralised *product policy* (line 8 of Algorithm 1 on page 75).

Segment relevance The *central authority* abhors the deletion of (perhaps ultimately useful) experience, and maintains a comprehensive map in a central repository, while selecting in Section 6.1.2 only highly relevant subsets of this privileged *graph representation* for deployment to robots in the field in order to handle storage constraints (c.f. Section 2.4). While no data fusion of landmarks takes place, the relevance of an incoming segment can be used by the *download policy* (c.f. Policy 5.2.1) to omit the incoming segment from the

privileged *graph representation* and can be measured by (a) an initial seed estimate of location (c.f. FAB-MAP, Section 2.1.3), and (b) topometric searches in modest graph neighbourhoods using the *search tool* (c.f. Section 4.1.2), and (c) frame-to-frame matching using VO (c.f. Section 2.1.1) as discussed in the implementation of the *commutation policy* (c.f. Section 4.4.1).

Here, as in Section 5.4, the *central authority* (as a powerfully equipped, time-insensitive *buyer*), uses the *ranking tool* to attempt matches to all LHS (insular) *node representations* (no ranking is performed), and subordinates (as *sellers*) use the *throttling tool* to send all RHS (foreign) content (no throttling is applied).

6.1.2 Correlated Checkouts

In contrast, during "check out" for the centralised coordination of tasks (c.f. Figure 5.3, right) subordinate agents act as *buyers* while the *central authority* acts as the sole *seller*. Here, the behaviour of the processing blocks discussed in Section 5.2.1 is inverted along with this change in roles.

Role inversion Principally, subordinates (as *buyers*) use the *sampling tool* to request exchanges relevant to the imminent processing of the upcoming dataset log (c.f. Definition 2.3.1), while the *central authority* (as *seller*) uses the *throttling tool* to stem network traffic according to these requests. We now discuss two throttling mechanisms which improve upon the whole-database deployment ("copy") discussed in Section 5.2.2.

Selection near the foray time of day ("tod") A simple strategy for selecting the most relevant portions of $\mathcal{G}_{\circledcirc}$ involves excluding any *node representations* outside of a temporal window, $t_w > 0$, around the time at which the mission commences, $0 < t_s < 1440\,\text{min}$ (24 h modulus), such that

$$\mathbb{T}(p,\ {}^{\overset{a}{\Xi}}\mathbf{P}^{\overset{s}{\Xi}}) = \begin{cases} 1, & \text{if } |\Upsilon_{\text{tod}}(v) - t_s| \leq t_w \\ 0, & \text{otherwise} \end{cases} \tag{6.1}$$

where t_s is the average time for the sample $({}^{\overset{a}{\Xi}}\mathbf{P}^{\overset{s}{\Xi}})$ as sent by the subordinate, calculated with the clock-synchronised time *node representation*, Υ_{tod} (c.f. Section 4.3.1) and the temporal window is typically set to three hours ($t_w = 180\,\text{min}$). Using the *throttling tool* in this way makes the assumption that appearance change is driven solely by the passage of time. This may account fairly well for illumination effects, but will fail on longer timescales in the face of seasonal and structural variation. This approach also relies on wide scope in the data available: experiences are required at all points during the day to cover the space of possible mission times. This naïve selection by time alone is promoted by further considering patterns within the use that the *canonical DM client* makes of the privileged *graph representation*, and inferring the set of nodes most likely to be localised against together.

Epochs As shown in Figure 6.2, the privileged *graph representation* is equipped with a record of successful localisations in the form of "Path Memory", a topology connecting nodes with paths, $\mathcal{P}_i$ (c.f. Section 4.2). Indeed, at this point, $\mathcal{G}_{\circledcirc}$ has been populated by forays as executed by members of

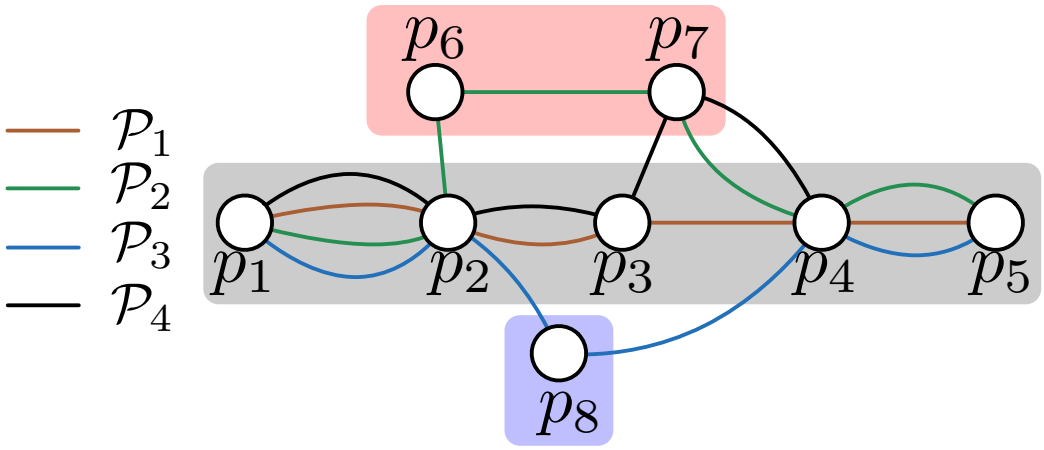


Figure 6.2: During normal mapping activity (nodes added on three separate occasions through the action of *patch representations* which are indicated by shaded red, grey, and blue regions), the EBN core records the use it makes of database nodes in the form of some uniquely identified paths through the topology. We can use a *path weight matrix* to recover the sequences of nodes most statistically relevant to a particular mission time. Note that the *patch elements* indicated carry odometry, metric localisation, and loop closures edges, as illustrated in Figure 6.1 on page 163, and these "Path Memory" statistics (brown, green, blue, and black) are stored as a *node representation*, $\Upsilon_{\text{paths}}(v)$ (c.f. Section 4.2) where the first *patch element* is registered to all paths registered to "Path Memory", $\mathcal{P}$.

the *fleet* driving around the world at times which fall into discrete windows of time throughout the day or "epochs"

$$\mathbf{e} = \{e_i = i \times t_w, 0 < i < \frac{1440}{t_w}, i \in \mathbb{R}^+\} \tag{6.2}$$

each of which are attached to the start of a sliding temporal window as discussed above.

Example 6.1.1. *In Figure 6.2 forays have been grouped into three epochs e_1, e_2, and e_3, which are shown grey, red, and blue, respectively. An agent which*

is about to embark on a foray which falls in the first epoch, e_1, will find nodes which are on the first path, $\mathcal{P}_1$, to be most useful as it never touches nodes outside of this epoch.

Path memory matrix We follow a simple counting procedure to keep track of which paths are registered against all *node representations* and populate a *path memory matrix* as

$$\mathbf{M}(v_i,\ \mathcal{P}_j) = \begin{cases} 1, & \text{if } \mathcal{P}_j \in \Upsilon_{\text{paths}}(v_i) \\ 0, & \text{otherwise} \end{cases} \tag{6.3}$$

Figure 6.2 illustrates how nodes can be registered to more than one path (e.g. v_2 is on all four paths).

Path count matrix Algorithm 3 transforms this *path memory matrix* to encode the co-occurence of paths by counting the total number of nodes (line 2) that each path touches within an e (line 1).

This transformation is inspired by motifs in biological sequences (such as genetic sentences) [Stormo et al., 1982], and we refer to it as the *path count matrix*, $\mathbf{C}$ (line 6).

Path weight matrix The statistical significance of a path at a certain time of day is inspected by normalising the *path count matrix* to obtain a *path weight matrix* as

$$\mathbf{W}(\mathcal{P}_i,\ e_j) = \frac{\mathbf{C}(\mathcal{P}_i,\ e_j)}{\sum_{\mathcal{P}_i \in \mathcal{P}} \mathbf{C}(\mathcal{P}_i, e_j)} \tag{6.4}$$

Algorithm 3 Counting nodes which originate from imagery within discrete epochs. If these nodes have been used in localisation along a particular path, we increment the usage of that path within the epoch.

Require: $\mathbf{e}$, $\mathcal{G}_{\circledcirc}$, $\mathcal{P}$
Ensure: $\mathbf{C}$

```
for e_n, e_{n+1} ∈ e do                                  ▷ Consecutive epoch pairs
    V_RADIATE ← {v ∈ G_⊚, e_n < ϒ_tod(v) < e_{n+1}}      ▷ Epoch nodes
    for v_i ∈ V_RADIATE do
        for P_j ∈ P do                                   ▷ "Path Memory"
            if M(v_i, P_j) = 1 then
                C(P_j, e_n) ← C(P_j, e_n) + 1
            end if
        end for
    end for
end for
```

which ensures that the values in each column sum to one (such that there is always a path registered to each epoch).

Example 6.1.2. *In the example illustrated in Figure 6.2, the path memory matrix can be enumerated, transformed into a path count matrix, and weighted to yield the path weight matrix as*

$$\mathbf{M} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \quad \mathbf{C} = \begin{bmatrix} 5 & 0 & 0 \\ 4 & 2 & 0 \\ 4 & 0 & 1 \\ 4 & 1 & 0 \end{bmatrix}, \quad \mathbf{W} = \begin{bmatrix} 0.28 & 0 & 0 \\ 0.24 & 0.67 & 0 \\ 0.24 & 0 & 1 \\ 0.24 & 0.33 & 0 \end{bmatrix} \tag{6.5}$$

using Equation 6.3, Algorithm 3, and Equation 6.4.

The dimensions of the *path count matrix* and *path weight matrix* are constant in the number of columns (the discretised epochs are preset) but not constant in the number of rows (which grows every time the *central authority* localises within its *graph representation*).

Discovering the most statistically relevant paths ("paths") Choosing the best paths to download before a foray involves looking for the most commonly used paths at times corresponding to the foray which are weighted with at least λ % as

$$\mathcal{P}_s = \{\mathcal{P}_i \in \mathcal{P}, \mathbf{W}(\mathcal{P}_i, e_s) > \lambda\},\ e_s = \underset{e \in \mathbf{e}}{\operatorname{argmin}}(e_i - t_s) \tag{6.6}$$

where e_s is the epoch closest to the mission time as specified by the sampled patch sent by the subordinate as a request, t_s (c.f. Equation 6.1) and $\lambda = 0.5$ typically (50 %).

Example 6.1.3. *Note (by inspecting the first column of the path weight matrix in Example 6.1.2) that $\mathcal{P}_1$ is less discriminative than the other paths in the "Path Memory" topology, as it more frequently co-occurs with all other paths.*

This use of the *throttling tool* selects the most statistically relevant paths for checking out just prior to a foray as a prior on the similarity in appearance

for the environment that we might expect before commencing the foray.

$$\mathbb{T}(p,\ {}^{\underline{a}}_{\Xi}\mathbf{P}^{\underline{s}}_{\Xi}) = \begin{cases} 1, & \text{if } \exists\ \mathcal{P}_i \in \Upsilon_{\text{paths}}(v), \mathcal{P}_i \in \mathcal{P}_s \\ 0, & \text{otherwise} \end{cases} \tag{6.7}$$

This is intuitive in that paths tacitly represent the relationship between experiences captured in the *graph representation*, where (for example) nodes storing sunny experience tend to be localised against during the same foray, and likewise for other broad environmental categories (e.g. snow, overcast, rain, etc.). More coarsely approximate techniques which filter for the "latest" *node representations* added to the privileged *graph representation* or predictive techniques which encode the temporal nature of scene elements (c.f. Section 2.2.4) and filter for the *node representations* which are commonly localised to by views in which those scene elements frequently occur could alternatively find expression in the *throttling tool* and the "checkout" routine in general.

6.1.3 Throttling and Ranking by Value

In the decentralised coordination of tasks (c.f. Section 5.2.3) agents may simultaneously act as *buyer* and/or *seller*. Thus the *ranking tool* and *throttling tool* are implemented by considering relative measures of value between exchanged *patch representations*, as there is no privileged *graph representation* over which historical localisation statistics can be computed as in Section 6.1.2.

Throttling Specifically, the *choice tool* (c.f. Tool 4.4.1) is used (in advance of its use for the *commutation policy*, c.f. Algorithm 2) to select the RHS content which is likely to replace or be superseded by some LHS content as

$$\mathbb{T}(p,\ {}^{\stackrel{a}{\Xi}}\mathbf{P}^{\stackrel{s}{\Xi}}) = \begin{cases} 1, & \text{if } \exists\ p_i \in\ {}^{\stackrel{a}{\Xi}}\mathbf{P}^{\stackrel{s}{\Xi}},\ \gamma(v, v_i) \in \{v, v_i\} \\ 0, & \text{otherwise} \end{cases} \tag{6.8}$$

which condition is equivalent to the *choice tool* strictly choosing between closely matching frames (i.e. indecision, $v_?$, is not allowed) and where $p = (\iota, g)\ \in\ {}^{\stackrel{b}{\Xi}}\mathbf{P}^{\stackrel{u}{\Xi}},\ g = (v, æ)$ are the candidate RHS patch elements for communication as per Equation 5.3 on page 132.

Ranking Similarly, the *choice tool* is used again to select the LHS content which is likely to supersede or be superseded by the incoming, throttled RHS content as

$$\mathbb{L}(p,\ {}^{\stackrel{b}{\Xi}}\mathbf{P}^{\stackrel{t}{\Xi}}) = \begin{cases} 1, & \text{if } \exists\ p_i \in\ {}^{\stackrel{b}{\Xi}}\mathbf{P}^{\stackrel{t}{\Xi}},\ \gamma(v, v_i) \in \{v, v_i\} \\ 0, & \text{otherwise} \end{cases} \tag{6.9}$$

where $p = (\iota, g)\ \in\ {}^{\stackrel{a}{\Xi}}\mathbf{P}^{\stackrel{u}{\Xi}},\ g = (v, æ)$ are the candidate LHS patch elements for matching as per Equation 5.4 on page 133.

Value Embedding the *choice tool* within the *throttling tool* and *ranking tool* implementation detail in this way is used in Section 6.2.1 to characterise a "price" for a data exchange which agents use as a persistent measure of the expected relevance of a peer's recorded navigation data to their own, which

is preconfigured in Section 6.3 by a guided assignment of dataset logs for processing to leading value providers.

Realtime evaluation Additionally, the complexity of evaluating all pairs of LHS-RHS *node representations* through the *choice tool* does not allow for the *patch representation* identified by the *sampling tool* ($\overset{a}{\Xi}\mathbf{P}\overset{s}{\Xi}$) to be used for selecting a likely peer to trade with to improve upon the explorative *sales policy* (c.f. Section 5.1). To this end, Section 6.2.2 discusses how DM agents randomly sample from $\overset{a}{\Xi}\mathbf{P}\overset{s}{\Xi}$ to estimate the realtime relevance of a peer's *node representations* in a tender-adjudication style process.

However, Equations 6.8 and 6.9 may still aggravate use of the CPU and network when LHS (for ranking) and RHS (for throttling) *patch representations* are not delimited by the *product policy* (c.f. Section 3.4.1). Section 6.1.4 thus discusses next the *shopping policy*, which agents use to focus requests to specific regions of the world.

6.1.4 Targeted Shopping Lists

Restricting the scope of the *diff tool* for patch delineation (c.f. Equation 3.25 on page 79) is a good mechanism for restricting the network, CPU and storage constraints that each agent may be burdened with. To this end, knowing (at least coarsely) location in the *product* space (c.f. Tool 3.4.1) is a prerequisite to deciding which part of the *product* space to explore.

Policy 6.1.1. *DM agents explore the sections of the world, $\mho$, understood by the product policy, $\wp$, by requesting delineated patch representations from*

their peers which have been assigned to a query product

$$\rho_{\text{QUERY}} = \zeta(\wp, \rho_{\text{LIVE}}), \ \rho_{\text{QUERY}} \in \mho \tag{6.10}$$

which the shopping policy, ζ, sets by considering the product, ρ_{LIVE}, which the robot is currently localised within by the placement tool.

Current The simplest shopping strategy identifies the current *product*, such that $\zeta_{\text{CURRENT}}(\wp, \rho_{\text{LIVE}}) = \rho_{\text{LIVE}}$. This is a sound strategy as agents will be most concerned with the region of the world that they are trying to localise within (to a precision finer than the *placement tool* offers).

Window However, robots are equipped with a variety of noisy sensors, and a more robust *shopping policy* might entail querying for *products* which are proximal in the *product* space, $\zeta_{\text{WINDOW}}(\wp, \rho_{\text{LIVE}}) \in \{\rho_{\text{PREVIOUS}}, \ \rho_{\text{NEXT}}\}$, which correspond to connected *products* in the topology shown in Figure 3.7 on page 73, and which should go some way to accounting for variation in the metric understanding of the world across the *fleet* (mitigating the effect of slightly inaccurate submaps).

Recommend Each *buyer* has access to their favourite *seller* for a particular *product* (c.f. *acquisitions tool*, Tool 3.4.3) as well as their own best-selling *products* (c.f. *stock tool*, Tool 3.4.4), by counting the number of exchanged *node representations* during uni- or bi-directional trading opportunities (c.f. Section 5.1). Following this intuition, this *shopping policy* thus employs a recommendation style decision-making process to occasionally shop

for the best-selling *product* from an overall favourite *seller* for the *product* that the *placement tool* has located the *buyer* within, $\zeta_{\text{RECOMMEND}}(\wp,\ \rho_{\text{LIVE}}) = \rho_{\text{RECOMMEND}}$.

Example 6.1.4. *An example of a recommended purchase is shown in Figure 3.7 on page 73, where the suggested product (ρ_2) traverses (in the opposite direction) the same route as the product that the placement tool locks on to (ρ_1), but has a measured relevance through the throttling tool and ranking tool (c.f. Equations 6.8 and 6.9) which arises due to loop closures between places in this region of the world.*

To facilitate this, Section 6.2.1 characterises these measured (by $\mathbb{T}$ and $\mathbb{L}$) effects upon the *graph representation* and Section 6.2.2 presents a persistent/anticipated "trust" factor which DM agents maintain for every identifiable team member ($\mathfrak{R}$), specific to a *product* in the catalogue ($\mathfrak{V}$), and which may be shared with one another as advisories.

6.2 Targeting Providers

This section characterises the relative value between exchanged *patch representations* of Section 6.1.3 as a (post-exchange) measured effect upon the insular *graph representations* for *buyers* as well as a bidding strategy which agents use to anticipate these effects prior to executing a data exchange with a *seller* chosen to maximise pairwise compatibility and localisation outlook.

6.2.1 Pricing Data Transactions

The *cost* of a transaction is proportional to the number of *node representations* throttled by the *seller* and communicated over the network, and is thus related to Equation 6.8 as

$$\kappa = \sum_{p \in {}^{\overset{b}{\Xi}}\mathbf{P}^{\overset{u}{\Xi}}} \mathbb{T}(p,\ {}^{\overset{a}{\Xi}}\mathbf{P}^{\overset{s}{\Xi}}),\ 0 \leqslant \kappa \leqslant |{}^{\overset{b}{\Xi}}\mathbf{P}^{\overset{u}{\Xi}}| \tag{6.11}$$

whereas the *reward* inherent to a transaction is proportional to the number of *node representations* ranked for matching and is related to Equation 6.9 as

$$\delta = \sum_{p \in {}^{\overset{a}{\Xi}}\mathbf{P}^{\overset{u}{\Xi}}} \mathbb{L}(p,\ {}^{\overset{b}{\Xi}}\mathbf{P}^{\overset{t}{\Xi}}),\ 0 \leqslant \delta \leqslant |{}^{\overset{a}{\Xi}}\mathbf{P}^{\overset{u}{\Xi}}| \tag{6.12}$$

These measurements must be considered in tandem in order for agents to guide their transactions by the competing concerns of localisation performance and resource management (c.f. Section 2.4).

Tool 6.2.1. *Thus, on applying a patch received from a seller, r_j, at time step k[1], the buyer, r_i, uses the pricing tool, Λ, to take a measurement of the value of the transaction as*

$$\eth_{i,j,k} = \Lambda(\kappa, \delta) = \frac{1}{1 + \exp(-\frac{\delta}{\kappa})},\ 0 \leqslant \eth_{i,j,k} \leqslant 1 \tag{6.13}$$

where the reward-cost ratio is normalised by a logistic function [Verhulst, 2009]

[1]To be clear, at this point k refers to the kth data exchange event between these agents.

to express a per-packet[2] *(node representation) "price".*

Here, the *choice tool* must leverage aspects of the *node representation* which correlate strongly with visual appearance.

Example 6.2.1. *Consider an implementation for the choice tool which uses the number of inliers in ego-motion estimation as the quality of representation for commutation policy, ranking tool, and throttling tool*

$$\gamma(v_i, v_j) = \begin{cases} v_i, & \textit{if } |\Upsilon_{\textit{inliers}}(v_i)| > |\Upsilon_{\textit{inliers}}(v_j)| \\ v_j, & \textit{otherwise} \end{cases} \tag{6.14}$$

where this node representation is a variant of the sparse feature points (Υ_{FAST} and Υ_{BRIEF}) in the compliant EBN graph representation of Section 4.3.1.

Valuing representations as in Example 6.2.1 proves in the experiments of Section 6.4.8 to not be amenable to the bandit games of Section 6.2.3 in that the features used for robustly estimating motion from video sequences are not necessarily suitable for recognising places with severe appearance change.

Category compatibility Alternatively, as mentioned in Section 4.4.2, the *choice tool* as implemented in this thesis can be intuited as choosing the *node representation* that is most well-explained by the FAB-MAP model when compared against frames, $\mathcal{L}_{\text{QUERY}}$, assigned to the query product, ρ_{QUERY}

[2]The cost and reward are limited to the size of exchanged patches as per Equations 6.11 and 6.12, while Equation 6.13 ensures that the price is well-conditioned in the range 0 to 1 , as there is no guarantee on the relative sizes of versioned representations, especially after agents have not interacted for some time.

(c.f. Section 6.1.4) by the *category tool* (c.f. Tool 3.4.2) as

$$\gamma(v_i, v_j) = v_i \leftrightarrow \sum_{v \in \mathcal{L}_{\text{QUERY}}} \mathbf{b}_i \cdot \mathbf{b} < \sum_{v \in \mathcal{L}_{\text{QUERY}}} \mathbf{b}_j \cdot \mathbf{b} \tag{6.15}$$

which requires the BoW (c.f. Section 4.3.1) *node representation*, Υ_{BoW}, and is illustrated here with cosine similarities (c.f. Equation 2.9 on page 25) but is in fact more efficiently implemented using FAB-MAP's optimal tree-structured Bayesian network approximation to the joint distribution over word occurrence [Cummins and Newman, 2011].

Under such a valuation framework, and if *sellers* are prohibited from forwarding foreign content (c.f. Equation 3.27 on page 80), a historical record of the measurements taken by the *pricing tool* is used next in anticipating the relevance of a data exchange with another member of the team for some region of the *product* catalogue.

6.2.2 Tracking Trust in Vendors

Equipped with this concept of value provision, each DM agent manages an internal belief in the expected price for performing a transaction with another agent, given the opportunity. To this end, a record of these measurements

$$\mathfrak{M}_{i,j} = \{\Im_{i,j,1}, \Im_{i,j,2}, \ldots\} \tag{6.16}$$

is stored by the *buyer*, r_i, as a reference for each potential *seller*, r_j.

Tool 6.2.2. *DM agents use a belief tool to model the evolving set of trade*

values specific to another identifiable agent

$$\mu_{i,j,k} = \frac{1}{k}\sum_{n=1}^{k-1} \eth_{i,j,n},\ \sigma_{i,j,k} = \sqrt{\frac{\sum_{n=1}^{k-1}(\eth_{i,j,n} - \mu_{i,j,k})^2}{k}} \tag{6.17}$$

as a normally distributed random variable, $\eth_{i,j,k} \sim \mathcal{N}(\mu_{i,j,k}, \sigma_{i,j,k})$*, the parameters for which can be shared with peers as advisories through the middleware technology discussed in Section 5.3.*

To track the average value being provided by this *seller* we use the numerical method suggested in [Knuth, 2007] to avoid the repeated (as well as increasingly imprecise) computation over all historical measurements of Equation 6.17, such that the sample mean is updated as

$$\mu_{i,j,k} = \mu_{i,j,k-1} + \frac{1}{k}(\eth_{i,j,k} - \mu_{i,j,k-1}) \tag{6.18}$$

while the sample standard deviation is indirectly tracked through the sample variance as

$$k\sigma^2_{i,j,k} = (k-1)\sigma^2_{i,j,k-1} + (\eth_{i,j,k} - \mu_{i,j,k-1})(\eth_{i,j,k} - \mu_{i,j,k}) \tag{6.19}$$

which are typically initially set to $\mu_{i,j,1} = 0$ and $\sigma^2_{i,j,1} = 1$ and where the latest measurement, $\eth_{i,j,k}$, is used as an observation to update these empirical quantities.

Anticipating value The record of measurements, $\mathfrak{M}$, used by the *belief tool* is updated as described above only after the full exchange has been

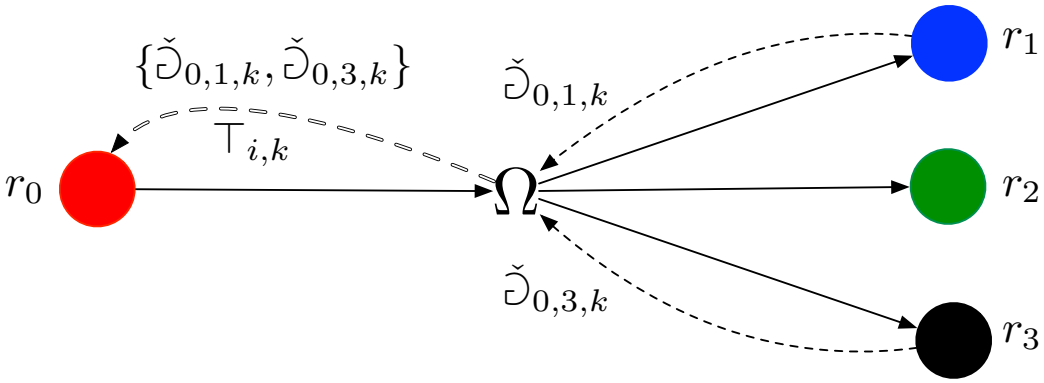


Figure 6.3: DM agents follow a tender-adjudication procedure in which a subset of their novel content (solid black lines) is sent to each available *fleet* member (where in this illustration r_2 is offline and does not respond), each of which performs a value measurement (dashed black lines) by comparing the sample to their own maps. In the greediest approach, the *buyer* then chooses to perform a full exchange with the *seller* for which measurement is least perturbed from the querying robot's internal beliefs.

completed (i.e. a processing block procession $\mathbb{S} \to \mathbb{L} \to \mathbb{V}$ from the *buyer's* perspective as illustrated in Figure 5.3). Figure 6.3 illustrates a tender-adjudication style pre-exchange handshaking procedure (executed by the methods discussed in Section 5.3.3) in which agents who are available to trade take the liberty of performing measurements, $\check{\mathfrak{I}}_{i,j,k} = \Lambda(\check{\kappa}_{i,j,k}, \check{\delta}_{i,j,k})$, on some (randomly chosen) subset of the sampled patch sent as a request by the *buyer*, which the *buyer* collects as "tender" offers

$$\top_{i,k} = \{(r_j, \check{\mathfrak{I}}_{i,j,k}) \mid r_j \in \mathfrak{R},\ j \neq i\} \tag{6.20}$$

which data structure consists of pairs of team member identifiers and real-valued *pricing tool* measurements of the relative value of the participating agents *patch representations*. These "tender offers" are used in Section 6.2.3 to bias the selection of trading partner when considering the history of ex-

changes as well as this realtime relevance.

6.2.3 Choosing Trades with Discrimination

The tender-adjudication process described in Section 6.2.2 is treated as an observation of the measured price before playing a lever corresponding to each *fleet* member in a multi-armed bandit stochastic scheduling scenario [Auer et al., 2002]. After k trades with a team member (before a possible trade at time step $k+1$), *buyers* therefore try to minimise

$$\sum_{n=1}^{k} E[|\check{\eth}_{i,j,n} - \eth_{i,j,n}|] \tag{6.21}$$

as the expected difference between value as captured (pre-exchange) by the "tender offer" and value as measured (post-exchange) by the *belief tool*, which guards against sudden perturbations in the value that *vendors* are typically providing.

Exploring value The decentralised experiments of Sections 5.4.5 and 5.4.6 used one such *sales policy*, $\Omega_{\text{bandit-explore}}$, which chose as trading partner a random member of the *fleet*.

Exploiting value This random selection was shown to scale with the size of the *fleet* more gracefully than the *distributed-query-all* approach, Ω_{dqa}, used to maximise localisation outlook in Section 5.4.4 and similarly to the centralised trading combination of *sales policies*, $\Omega_{\text{dedicated}}$ and Ω_{central}, in Sections 5.4.1 to 5.4.3. In the simplest discriminative case, however, the

buyer (r_i) uses a *distributed-query-best* strategy to choose the *seller* for which the "tender offer" is least perturbed from its internal beliefs, such that

$$\Omega_{\text{bandit-exploit}}(\mathfrak{R},\ k) = \underset{r_j\ \in\ \mathfrak{R}, j\neq i}{\operatorname{argmin}} |\check{\eth}_{i,j,k} - \eth_{i,j,k}| \tag{6.22}$$

which as a *sales policy* is a purely exploitative decision-making process suited for situations in which a particular agent consistently provides higher quality content from one part of the environment (e.g. it has mapped that environment more regularly).

Bandit games However, the world is vast, its appearance is subject to complex modes of change, and in lieu of a principled assignment of dataset logs (as will be discussed in Section 6.3 below) there is no guarantee that an agent exploited for highly relevant *patch representations* at one trading opportunity will provide the same level of value at further communication events. We thus apply an *epsilon-greedy* approach, $\Omega_{\text{bandit-explore-exploit}}$, to the selection of trading partner, either exploiting the best lever or exploring for fractions ϵ and $1-\epsilon$ [Langford and Zhang, 2008] of the total number of trading events respectively, where a typical value of $\epsilon = 0.5$ is tuned for the FAB-MAP compatibility approach in Section 6.4.6 below.

The tender-adjudication process illustrated in Figure 6.3 lends itself well to alternative treatments of the regret in Equation 6.21, where this epsilon-greedy approach is limited in that it does not consider upper confidence bound (UCB) estimates for the candidate lever distributions for reducing uncertainty in the knowledge gained for each candidate [Sofman et al., 2010].

6.3 Assigning Leading Value Providers

This section concludes the chapter by presenting a method for pre-assigning dataset logs (c.f. Sections 2.3.2 and 2.3.3) to agents in the MAS experiments which follow.

Guiding motivation This procedure is typically not undertaken (e.g. in all experiments undertaken in Chapters 4 and 5 and all experiments in Section 6.4 except Section 6.4.9), but is useful in proving the validity of the bandit game approach to maximising the relative value (i.e. pairwise matches discovered in branching model manipulations as well as localisation outlook) gleaned from data exchange events.

Heatmap construction Consider the total set of dataset logs (c.f. Section 2.3 and Appendix A) available, $\mathcal{D}$. A Visual Similarity Matrix, $\mathbf{V}_i$, is constructed for each section of the world as understood by the *product policy*, ρ_i, using the cosine similarity (c.f. Equation 2.9 on page 25) between the bag of words for representative imagery from each log (c.f. Section 2.1.3).

Heatmap vectorisation Averaging the similarity terms in $\mathbf{V}_i$ for each of the logs, $\ell_j \in \mathcal{D}$, as compared to every other log, $\ell_n \in \mathcal{D}$, reduces this VSM to a vector

$$\mathbf{v}_{i,j} = \frac{1}{|\mathcal{D}|} \sum_{n=1}^{|\mathcal{D}|} \mathbf{V}_i(j, n) \tag{6.23}$$

the real-valued elements of which capture the aggregated visual similarity of one the dataset logs, ℓ_j, to all others (i.e. $|\mathbf{v}_{i,j}| = |\mathcal{D}|$) within a region

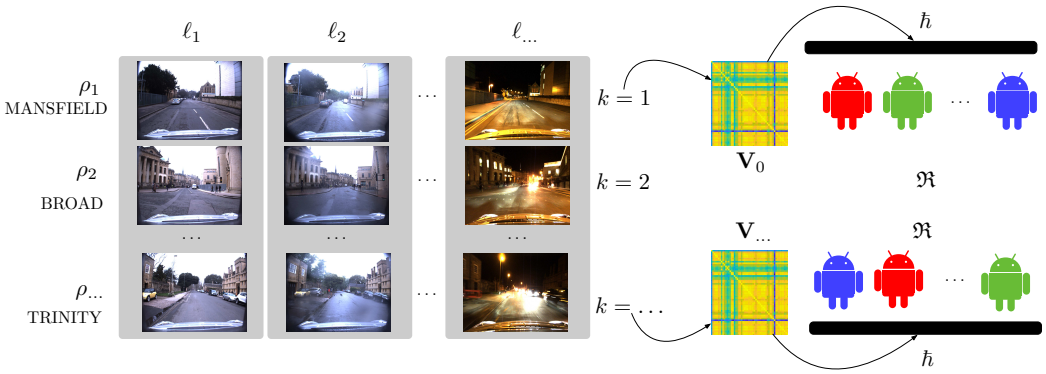


Figure 6.4: An expert algorithm which imposes organisational structure on imagery *before* assigning it to each agent for processing is used to consistently assign dataset logs (specifically, within a section of the world) which are most self-similar to an "leading" agent. In this figure, time steps k indicate that agents are traversing a new section of the world before each trade and after the "leading" agent changes to another member of the *fleet*, although the *heatmap tool* is flexible in this implementation detail.

specified by a *product*, ρ_i.

Tool 6.3.1. *This vector is manipulated by the heatmap tool, $\hbar$, as*

$$\hbar(\mathbf{v}_{i,j}, \ell_j) = r_n \in \mathfrak{R} \tag{6.24}$$

for assignment of the most visually similar set of logs to a "leader", r_n, in the fleet for the product, ρ_i.

This agent then becomes the leading value provider in that region of the world as it experiences and maps views of places in that region which are more likely to be registered against one another.

Example 6.3.1. *In Figure 6.4, the red agent is assigned leader in the* MANSFIELD *product category, while the blue agent is assigned leader for* TRINITY*. The first two dataset logs (ℓ_1 and ℓ_2) are notably more similar in appearance than the last.*

This intuition has precedent in [Churchill, 2012] in which it was shown that moving the most "interesting" or "surprising" forays to the beginning of plastic map creation resulted in the accumulation of experiences being high before dropping off significantly.

Heatmap assignment The vectorised VSM is typically rearranged by the *heatmap tool* according to the magnitude of its similarity scores in order to achieve (a) random assignment of logs across the *fleet*, or (b) assignment of the most visually similar logs to one *vendor*, or (c) a demarcated assignment, in which each *vendor* in the *fleet* takes on the role of leading value provider in a *product* category.

The simplicity of this empirical approximation of superior localisation based on visual similarity alone has precedent in [MacTavish et al., 2017], in which local visual vocabularies (constructed online) are used for experience-selection in maps built from imagery as distinct as night and day.

6.4 Results

This section specifically addresses "guiding question" item 4 on page 5, further managing the resourcing concerns discussed in Section 2.4 as emergent in the experiments presented in Section 5.4.

In particular, we apply the content and provider selection techniques presented in Sections 6.1 and 6.2 to the interaction of the processing blocks discussed in Section 5.2.1 and verify the applicability of the bandit game approach to bolstering both pairwise compatibility in branching model ma-

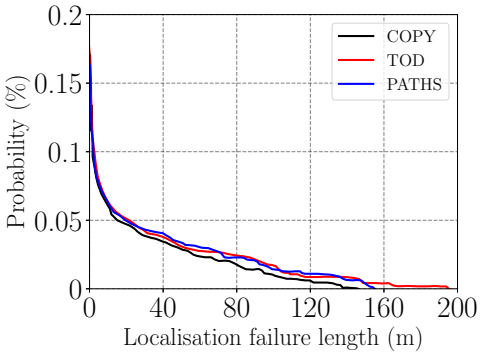
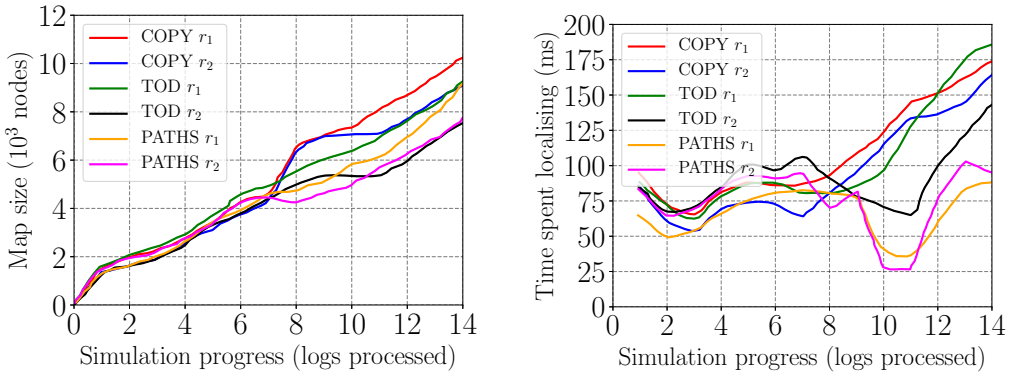


Figure 6.5: Cumulative failure distributions for correlated checkout techniques, aggregated over all agents and trials. Ideal localisation performance (COPY, black, with AUC 3.67, c.f. Section 2.3.1) is approximated coarsely by the selection of database content recorded at similar times during the day (TOD, red, 4.60) and more accurately by selection using the history of localisation success for the privileged *graph representation* (PATHS, blue, 4.47).

nipulations and localisation outlook while shrewdly managing the adverse CPU, disk, and network burdens.

Section 6.4.1 presents results which were obtained by 5-fold cross-validation (c.f. Section 2.3.2) over the BEGBROKE-24HR urban dataset (c.f. Appendix A.2.1, using the experience segment *product policy* of Section 3.4.1).

In contrast, Sections 6.4.2 to 6.4.4 and 6.4.6 to 6.4.9 aggregate statistics over randomised trials with no held-out sets (c.f. Section 2.3.3) using video sequences from thirteen of the *product* space regions specified by the GPS catalogue *product policy* over the *Oxford RobotCar Dataset* (c.f. Appendix A.2.2).



(a) (b)

Figure 6.6: Per-agent (r_1 and r_2) instrumentation for (a) cumulative database sizes and (b) average time spent localising for correlated checkout techniques. Whole-database *throttling* (COPY, red and blue) imposes monotonous growth throughout the lifetime of operation of each robot. In (a), filtering by time of day (TOD, green and black) and statistically relevant paths (PATHS, orange and pink) periodically (between finishing processing one dataset log group and starting another, around the eighth dataset log) subsamples the amount of information required to be stored by each agent's disk. In (b), distributing these smaller *graph representations* to agents in the field improves the rate that the *canonical DM client* operates at, where (e.g. upon processing dataset log ten within the second and final group), the average time (across all agents) improves from 80.5 ms (COPY, red and blue) and 80.1 ms (TOD, green and black) to 71.48 ms (PATHS, orange and pink).

6.4.1 Distributing Mission-Specific Maps

This experiment improves upon the whole-database ("copy") deployment of Section 5.4.3 by subsampling portions of the *central authority's* privileged *graph representation* for checking out by the methods described in Section 6.1.2. Here, two agents (r_1 and r_2) perform five cross-validation trials, each of which consists of a group of nine dataset logs which traverse the 800 m extent of the ρ_{BEGBROKE} *product*, and communicate (after processing each group) via the centralised trading configuration (c.f. Figure 5.1, left)

and a third agent which acts as the *central authority*.

In Figure 6.5 the time of day approach (TOD, red) results in a maximum localisation failure of 195.63 m, as compared to 155.48 m or 148.01 m when discovering the most statistically relevant paths (PATHS, blue) or using the idealised whole-database checkout (COPY, black) respectively. This is attributed to the omission of *node representations* which are localised against by two or more significantly different experiences, starving the *throttled* map of information-rich experience-to-experience information.

Additionally, Figure 6.5 shows, for a safe dead-reckoning driving distance of 80 m, that this implementation of the *throttling tool* reduces the probability of experiencing an unsafe failure from 2.41 % (TOD, red) to 2.25 % (PATHS, blue) as averaged over all dataset logs within the five held-out groups, approaching the idealised whole-database deployment of 1.73 %. This is attributed to the canny selection of only content relevant to the current foray (excluded database content will not have aided localisation).

In Figure 6.6(a), selecting by time of day (TOD, green and black) with a band of $t_w = 3\,\text{h}$ (180 min, c.f. Equation 6.1) around the current foray time (t_s, c.f. Equation 6.2) offers immediate respite, where each DM agent is required in between forays (around dataset log seven) to respectively store only 81.77 % and 79.57 % of the *node representations* it would be required to hold for a complete check out (COPY, red and blue). Additional filtering by path, choosing the top 50 % ($\lambda = 0.5$, c.f. Section 6.1.2), reduces the storage requirement even further – where the savings become 74.56 % and 66.57 % respectively (PATHS, orange and pink).

The disk loads (measured by node counts) for Figure 6.6(a) are quantified

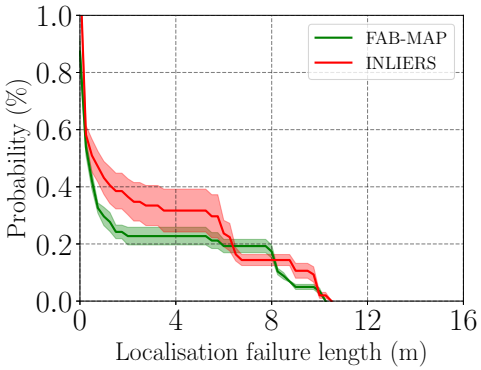
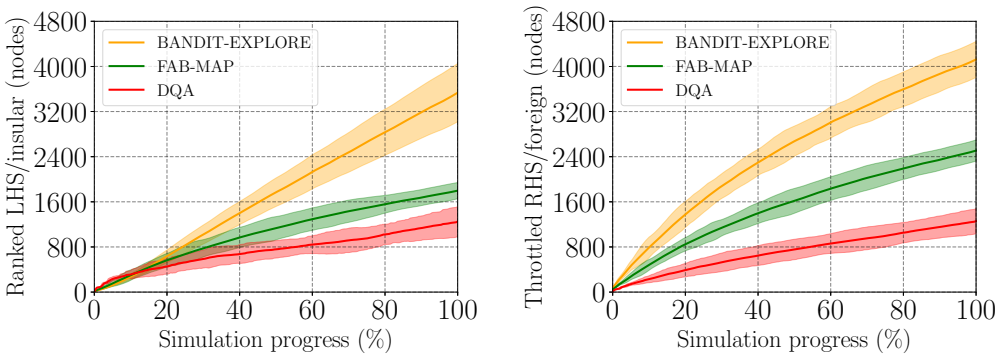


Figure 6.7: Cumulative failure distribution when embedding (through the *choice tool*) either FAB-MAP (green) or quality of ego-motion estimation (red) within the *throttling tool* and *ranking tool* implementations. For a safe dead-reckoning driving distance of 4 m, the FAB-MAP approach reduces the probability of experiencing an unsafe failure from 31.68 % ± 7.48 % to 22.74 % ± 3.08 %, whereas for a safe dead-reckoning driving distance of 7 m, the ego-motion approach reduces the probability of experiencing an unsafe failure from 19.26 % ± 2.33 % to 14.37 % ± 1.92 %.

as 6857 and 6506 (COPY, red and blue), 5359 and 5111 (TOD, green and black), and finally 4654 and 4524 (PATHS, orange and pink) – each for r_1 and r_2, respectively

This corresponds to a bulk transfer of data from the *central authority* which can be enumerated by the *patch representations* communicated as 160.62 kB and 158.61 kB (COPY), 131.56 kB and 122.51 kB (TOD), and finally 112.09 kB and 109.33 kB (PATHS) – each for r_1 and r_2, respectively.

Figure 6.6(b) compares the average time spent localising at points during each foray, where subsampling the privileged *graph representation* is more lenient on the CPU resources required due to a kinder throughput to FAB-MAP relocalisation (as the bottleneck in EBN performance). This is attributed to the localiser operating more efficiently as a greater proportion of the *throttled* map is relevant to the current mission.



(a) (b)

Figure 6.8: Cumulative number of nodes (a) ranked by the requesting agent for matching as input to the *commutation policy* and (b) throttled by the requested agent in response to the *node representations* identified by the *sampling tool*. Here, the *distributed-query-all sales policy* (red) in conjunction with no restrictions on the *throttling tool* or *ranking tool* does not result in ever-increasing LHS and RHS candidate sets. *Throttling* and *ranking* of the data exchange in conjunction with a *distributed-query-one sales policy*, however, does (orange). This is mitigated by considering for matching relative value between exchanged patches (green).

6.4.2 Valuing Data in the Market

In order to investigate the impact of restricting the scope of the *throttling tool* and *ranking tool* by embedding the *choice tool* as a value framework in the decentralised trading configuration (c.f. Equation 6.8 on page 171 and Equation 6.9 on page 171), this experiment follows that of Section 6.4.1 by demoting the *central authority* such that the MAS is populated by equally-equipped collaborating localisers (c.f. Section 5.2.3). Here, four agents engage in ten trials of ten forays each over a 20 m portion of the $\rho_{\text{OBSERVATORY}}$ *product* and use the *distributed-query-all sales policy* (Ω_{dqa}) to execute trades with every other agent at every opportunity (c.f. Section 5.1.2).

Figure 6.7 shows cumulative failure distributions with maximum failures

which are comparable to those observed in Section 5.4.4 for idealised (unrestricted) *throttling tool* and *ranking tool* implementations.

In the 0 m to 6 m range of localisation failures, fewer failures are experienced when FAB-MAP scores are used as a selection mechanism (green, c.f. "category compatibility in Section 6.2.1) as compared to consideration of the number of inliers yielded through ego-motion estimation using VO (red, c.f. "geometric refinement" in Section 4.2). However, in the 6 m to 8 m range of localisation failures, the opposite effect is observed. This suggests (and is explored further in Section 6.4.6 below) that appearance-based approaches are most suitable for optimising pairwise compatibility of exchanged *patch representations*.

6.4.3 Limiting Throughput by Value

This experiment presents auxiliary analysis for the results shown in Section 6.4.2 in order to verify the beneficial impact of the *throttling* and *ranking* of exchanged *patch representations* (c.f. Section 6.1.3) on the realtime CPU and bandwidth burdens. Here, Figure 6.8 summarises statistics over ten trials in which four agents engage in twenty forays over a 30 m portion of the ρ_{ORI} *product*.

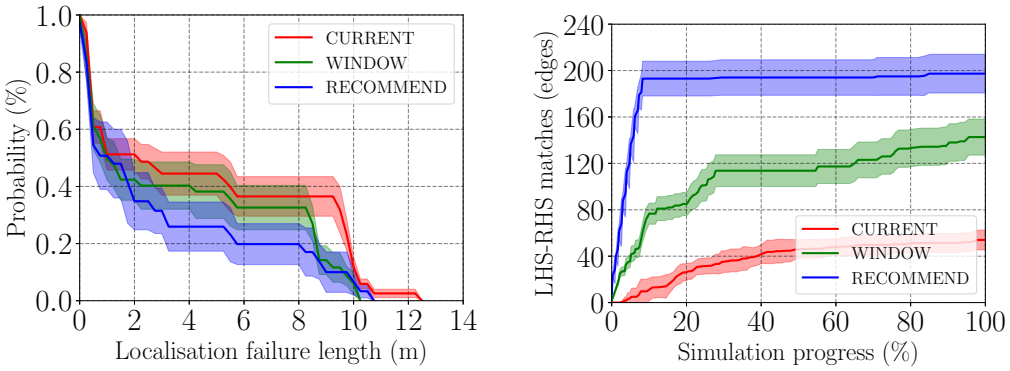
The *distributed-query-all sales policy* with no restrictions placed on the *throttling tool* or *ranking tool* is used as a baseline for comparison (DQA, red). Then, the *distributed-query-one sales policy* (c.f. Section 5.1.2) with no restrictions on the *throttling tool* or *ranking tool* (BANDIT-EXPLORE, blue) is compared to the same *sales policy* with restrictions dictated by the

choice tool as implemented with FAB-MAP (green).

Figure 6.8(a) shows the accumulated number of LHS (ranked) *node representations* used to match against incoming (throttled) *patch representations* by requesting agents. The unrestricted *distributed-query-all* approach actually induces the lightest load on the EBN localisation module (c.f. Section 4.2), as the size of the *patch representation* identified by the *diff tool* is kept somewhat constant due to exact synchronisation of agents' *history representations* (c.f. Figure 3.5(a)). This is in contrast to the unrestricted *distributed-query-one* (orange) approach, the CPU load for which reaches3535± 860 nodes as compared to 1457 ± 183 nodes (DQA, red), due to a divergence in the size of *state representations* (c.f. Figure 3.5(b)) as an undesirable effect of performing data exchanges with one peer at a time. *Ranking* insular *node representations* by their compatibility to the throttled content with FAB-MAP (green) mitigates this effect and approaches the idealised CPU load (1840 ± 59 nodes, green).

Likewise, Figure 6.8(b) shows the accumulated number of RHS (throttled) *node representations* published over MOOS by responding agents (c.f. Section 5.3.3), where *throttling* foreign *node representations* by their compatibility to the sampled content (c.f. *sampling tool*, Tool 5.2.1) sent by requesting agents Here, the idealised network load (red, 604 ± 54 nodes) is more closely approximated by targeting content (green, 1115 ± 42 nodes) rather than consuming all available content from one peer (orange, 3662 ± 141 nodes).

6.4.4 Managing a Useful Shopping List



(a) (b)

Figure 6.9: (a) Cumulative failure distributions in which the RECOMMEND approach reduces the probability of experiencing an unsafe failure of 5 m to 25.93 % ± 8.59 % from 44.49 % ± 7.54 % and 38.20 % ± 9.39 % for CURRENT and WINDOW approaches, respectively. (b) Accumulated number of pairwise matches over all data exchange events (i.e. the sum of all set sizes, $\mathcal{M}$, in Equation 4.18 on page 104).

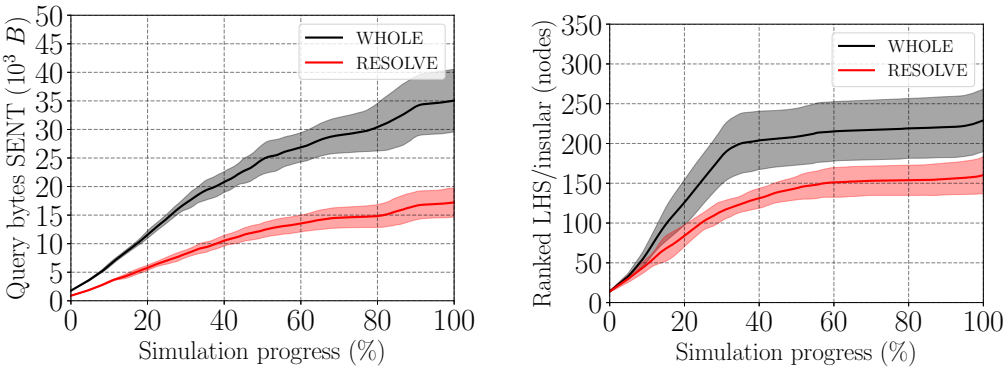
This experiment matures the decentralised system exhibited so far in Section 6.4.3 by applying the *shopping policy* (c.f. Section 6.1.4) as a mechanism for optimising the interaction of the *sampling tool*, *throttling tool*, and *ranking tool* (c.f. Section 5.2.1) within regions of the world identified by the itemised data exchange notation of Section 3.4.2.

Here, Figure 6.9 aggregates statistics over ten trials in which three agents engage in ten forays over a 90 m portion of the $\rho_{\text{MATERIALS}}$ *product* and use the *distributed-query-one sales policy* ($\Omega_{\text{bandit-explore}}$) to execute trades with a randomly selected peer at every opportunity (c.f. Section 5.1.2). the 90 m street adjacent to the MATERIALS building, passing by several distinctly-clad buildings, is demarcated in9 m increments to resolve the $\rho_{\text{MATERIALS}}$ *product* into ten contiguous *products* which each robot registers to its *product policy* and *category tool* (c.f. Tool 3.4.2).

Figure 6.9(a) indicates that the simplest strategy, purchasing the *product* currently (ζ_{CURRENT}, blue) placed within by the *placement tool* (c.f. Tool 3.4.1), is outperformed by including in the shopping list *products* in a WINDOW adjacent to CURRENT (ζ_{WINDOW}, green). The WINDOW strategy allows requesting agents to queue upcoming route sections before localisation of their live image streams, and results summarily in an AUC of 3.69 ± 1.61 as compared to 4.48 ± 1.65 (CURRENT, red). Both methods are outperformed by including *products* which other *team* members RECOMMEND ($\zeta_{\text{RECOMMEND}}$, blue), achieving an AUC of 2.87 ± 1.67. This recommendation strategy leverages the sizes of the sets represented by the *stock tool* (c.f. Tool 3.4.4, $\mathcal{S}$) and *acquisitions tool* (c.f. Tool 3.4.3, $\mathcal{A}$) to encourage requesting agents to occasionally redirect the query *product*, ρ_{QUERY}, which counteracts imperfectly aligned *products* from the GPS catalogue *product policy* in use (c.f. Section 3.4.2).

Indeed, Figure 6.9(b) shows how these *shopping policies* exhibit a greater number of (accumulated) pairwise matches (143 ± 16 and 197 ± 17 matches as compared to 54 ± 8 matches for WINDOW, CURRENT, and RECOMMEND respectively) over the course of many data exchange events, ultimately targeting trades which are likely to yield bonafide matches. This is once attributed to the facility for some flexibility in finding correspondences between foreign and insular content by measuring the effect of popularity on the *stock tool* and *acquisitions tool* (c.f. Section 6.1.4 for "best-selling" and "favourite seller").

6.4.5 Catalogue Tool Caching



(a) (b)

Figure 6.10: Cumulative (a) query *patch representations* sent over the network and (b) LHS *node representations* ranked for matching to foreign *patch representations*. In (a) network throughput is reduced from 35 064.80 B ± 5486.85 B to 21 683.34 B ± 3209.53 B by endowing agents with a more nuanced understanding of the space in which they are operating (RESOLVE, red). In (b) the *canonical DM client* must consider 229 ± 56 candidate nodes in total as compared to 160 ± 23 by restricting the full set registered to the *product policy* (WHOLE, black).

This experiment presents auxiliary analysis for the results shown in Section 6.4.4 in order to illustrate the boons to run-time resource management (in terms of CPU and bandwidth burdens) that a more nuanced understanding of the environment by resolving *products* into finer contiguous *products* (further motivating the *product policy* framework itself). Here, Figure 6.10 aggregates statistics over ten trials in which two agents engage in ten forays over a 40 m portion of the $\rho_{\text{ROYAL-OAK}}$ *product* (demarcated in5 m increments into eight contiguous *products*) and use the *distributed-query-all sales policy* to execute trades every other peer at every opportunity.

Figure 6.10(a) shows averages for the cumulative query *patch representations* sent over the network by requesting agents, where 21 683.34 B ± 3209.53 B as compared to 35 064.80 B ± 5486.85 B are required to complete

	EAGLE-AND-CHILD	OGILVIE
WHOLE	$23.68\,\text{kB} \pm 0.31\,\text{kB}$	$23.33\,\text{kB} \pm 0.58\,\text{kB}$
RESOLVE	$10.77\,\text{kB} \pm 0.12\,\text{kB}$	$9.40\,\text{kB} \pm 0.48\,\text{kB}$

Table 6.1: Improved accumulated network load as a result of demarcating data in the DM by use of the *product policy*, saving 45.47 % and 40.28 % (RESOLVE) as compared to the baseline (WHOLE) over the EAGLE-AND-CHILD and OGILVIE *product* spaces, respectively.

	EAGLE-AND-CHILD	OGILVIE
WHOLE	$58.36\,\text{s} \pm 4.95\,\text{s}$	$14.16\,\text{s} \pm 1.84\,\text{s}$
RESOLVE	$16.05\,\text{s} \pm 1.46\,\text{s}$	$4.27\,\text{s} \pm 0.74\,\text{s}$

Table 6.2: Improved accumulated CPU load in terms of the time dedicated by requesting agents to collecting the "tender offers" of Equation 6.20 on page 179 (in contrast to Figure 6.10(b)). Demarcating data in the DM by use of the *product policy* saves 27.50 % and 30.17 % (RESOLVE) as compared to the baseline (WHOLE) over the EAGLE-AND-CHILD and OGILVIE *product* spaces, respectively.

all data exchanges when the entire world (WHOLE, black) is resolved (RESOLVE, red) into *product* categories.

Likewise, Figure 6.10(b) shows a more lenient load on the localisation module of the *canonical DM client* (and consequently *commutation policy*) when only considering through the *ranking tool* candidate nodes in the *product* category involved in the query (ρ_{QUERY}, RESOLVE, red), as opposed to a candidate node set which encompasses all of the content which the agents involved in a trade have not yet compared (i.e. all *products*, $\mho$, WHOLE, black).

Tables 6.1 and 6.2 provide related statistical indication of this improvement in CPU and network load, respectively, where each entry corresponds to an experiment of ten trials in which two agents engage in ten forays over

	ORI
FAB-MAP-0.1	312 ± 24
FAB-MAP-0.3	363 ± 16
FAB-MAP-0.5	465 ± 14
FAB-MAP-0.7	387 ± 10
FAB-MAP-0.8	94 ± 4
INLIERS-0.1	141 ± 13
INLIERS-0.3	133 ± 5
INLIERS-0.5	146 ± 7
INLIERS-0.7	148 ± 12
INLIERS-0.8	149 ± 8

Table 6.3: Using an appearance-based metric within the *choice tool* implementation such as FAB-MAP shows better perturbation in performance (measured here as the number accumulated matches over all trades, i.e. the sum of all set sizes, $\mathcal{M}$, in Equation 4.18 on page 104) as the trial exploration ratio (c.f. Section 6.2.3) is varied, reaching a maximum of 465 ± 14 as compared to 149 ± 8 matches for trial exploration ratios of $\epsilon = 0.5$ (FAB-MAP) and $\epsilon = 0.8$ (INLIERS) respectively.

a 40 m portion of the $\rho_{\text{EAGLE-AND-CHILD}}$ and ρ_{OGILVIE} *products*.

6.4.6 Targeting Valuable Interactions

This experiment continues applying the FAB-MAP compatibility and quality of ego-motion *choice tool* implementations (used to limit throughput in Section 6.4.2) for optimising localisation outlook and the quantity of pairwise matches through the *commutation policy*.

To this end, we aggregate statistics over ten trials in which four agents engage in twenty forays over a 10 m portion of the ρ_{ORI} *product* and use the *epsilon-greedy sales policy*, $\Omega_{\text{bandit-explore-exploit}}$, to execute a trade with one other peer (c.f. Section 6.2.3). A varied trial exploration ratio, $\epsilon \in \{0.1, 0.3, 0.5, 0.7\}$, is used to examine the suitability of both approaches

within the bandit game setting discussed in this chapter.

Table 6.3 shows for the FAB-MAP compatibility approach greater variability with this parameter. Thus, while tuning localisation outlook with either approach was shown in Section 6.4.2 to be suitable for distinctly designed maximum localisation failure lengths, it is clear that the appearance-based (FAB-MAP) approach lends itself better to integration into *commutation policy* (c.f. line 18 of Algorithm 2 on page 105) in terms of the relevance of incoming *throttled* content.

6.4.7 Exploiting Trusted Vendors

This experiment uses the tuned FAB-MAP compatibility *choice tool* implementation of Section 6.4.6 with trial exploration ratio, $\epsilon = 0.5$, to investigate the behaviour of the various decentralised *sales policies* within the system.

Here, Figure 6.11 aggregates statistics over ten trials in which four agents engage in five forays over a 40 m portion of the $\rho_{\text{BACKPACKERS}}$ *product* using either the *distributed-query-all*, *distributed-query-one*, *distributed-query-best*, or *epsilon-greedy sales policies*.

The cumulative failure distributions of Figure 6.11 reveals the *epsilon-greedy* approach as promoting localisation performance which most closely approximates the idealised *distributed-query-all* AUC of 3.26 ± 1.22 (DQA, red) with4.86 ± 3.62 (BANDIT-EXPLORE-EXPLOIT, blue) as compared to 6.56 ± 2.79 and 9.27 ± 3.26 for the purely exploitative (BANDIT-EXPLOIT, green) and explorative techniques (BANDIT-EXPLORE, orange, respectively). Indeed, as Section 6.4.8 will analyse further, perturba-

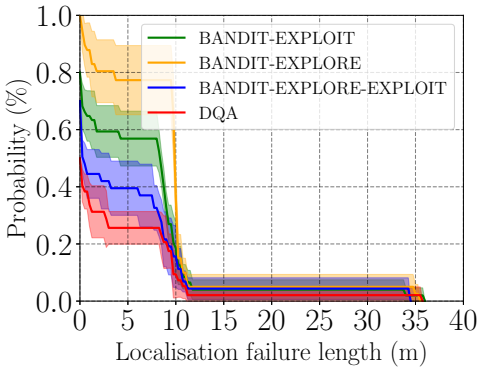


Figure 6.11: Cumulative distributions for all decentralised *sales policies*. For a safe dead-reckoning driving distance of 8 m, the *epsilon-greedy* approach most closely approximates the probability of experiencing an unsafe failure of 25.62 % ± 5.72 % (DQA, red) with32.62 % ± 7.84 % (BANDIT-EXPLORE-EXPLOIT, blue) as compared to 56.86 % ± 9.60 % (BANDIT-EXPLOIT, green) and 77.32 % ± 12.10 % (BANDIT-EXPLORE, orange) for the purely exploitative and purely explorative techniques, respectively.

tion in the choice of peer to send requests to is required in order to avoid a local maximum in appreciation for a member of the *team*, where data variability dictates that some other member will eventually enter data into the market that every other robot will find useful (e.g. ephemeral, information-rich appearance change such as irregular snow cover in Southern England).

Table 6.4 presents, as in Table 6.3, the accumulated matches discovered when consolidating divergent *patch representations*. Here, each entry corresponds to an experiment in which four agents engage in thirteen forays over a 10 m portion of the ρ_{PARADISE} or ρ_{BEAUMONT} *products*, from which it is clear that the explorative-exploitative hybrid approach is likewise better suited for tuning pairwise compatibility of *history representations*.

6.4.8 Asymmetric Beliefs

	PARADISE	BEAUMONT
BANDIT-EXPLORE	74 ± 12	30 ± 5
BANDIT-EXPLOIT	104 ± 39	50 ± 16
BANDIT-EXPLORE-EXPLOIT(100 %)	227 ± 29	124 ± 6

Table 6.4: The *epsilon-greedy sales policy* outperforms a purely explorative or purely exploitative approach and maximises the total number of matches discovered over all data exchange events. Here, alternative decentralised *sales policies* yield 45.54 % and 40.32 % (BANDIT-EXPLOIT) and 32.67 % and 24.19 % (BANDIT-EXPLORE) matches in total over two *products* registered to the *product policy*.

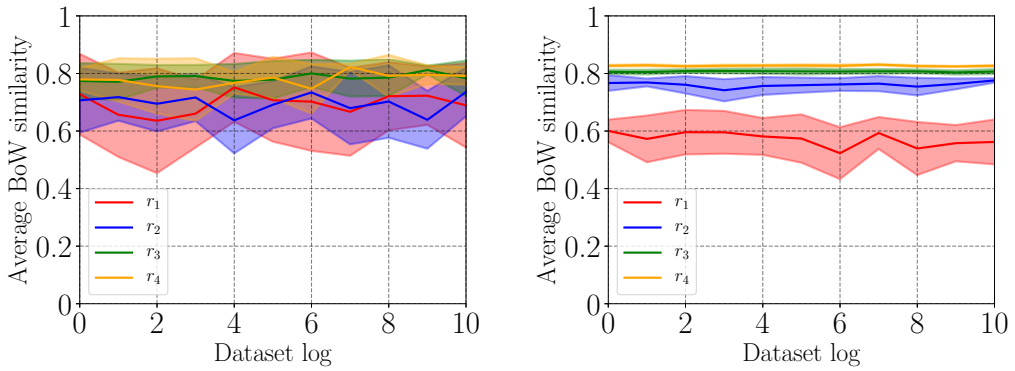


Figure 6.12: Average bag of words cosine similarity from each dataset log to all others as assigned (a) randomly or (b) greedily to each MAS collaborator. In (a) a random assignment is made in which each *team* member is responsible for data from a wide swathe of appearance conditions. In (b) a principled assignment is made in which *team* members are assigned successively more easily matchable data.

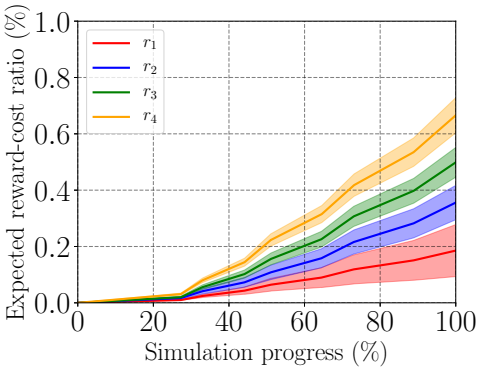
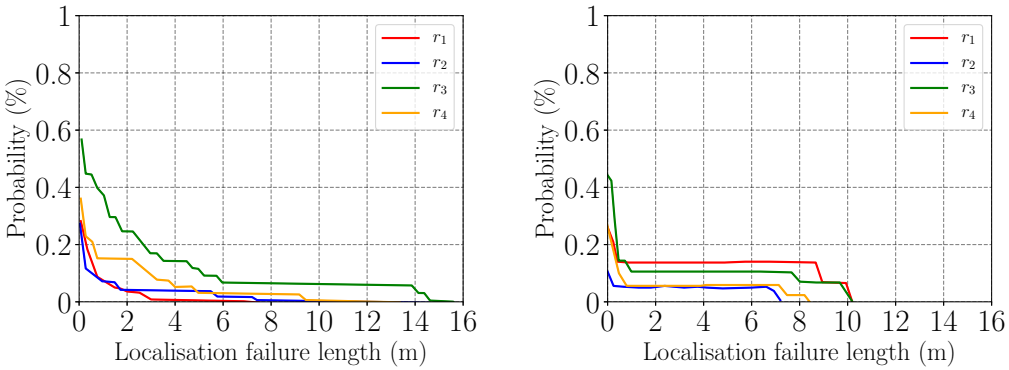


Figure 6.13: Aggregated tracked trust factors as (normalised by a logistic function, c.f. Equation 6.13 on page 175) ratios between the cost of an exchange (number of ranked LHS *node representations*) and the reward for an exchange (number of throttled RHS *node representations*), corresponding to the principled assignment of dataset logs in Figure 6.12(b).

This experiment uses the leading value provider assignment technique discussed in Section 6.3 to expose the evolution of the "trust factors" as recorded by each requesting agent's *belief tool* (c.f. Tool 6.2.2) during explorative and exploitative data exchange events in the coordination of trading tasks in experiments such as those presented in Section 6.4.7.

Figures 6.12 and 6.13 thus aggregate statistics over ten trials in which four agents engage in ten forays over a 10 m portion of the ρ_{HOLYWELL} *product* using the *epsilon-greedy sales policy* and a *choice tool* implementation which considers FAB-MAP compatibility between exchanged *patch representations*.

The *heatmap tool* (c.f. Tool 6.3.1, $\hbar$) is applied to the dataset logs to be processed by the *fleet* in order to assign imagery which is most similar in appearance to a hypothesised "average view" of this environment (randomly in Figure 6.12(a) and in the reverse order of the robot identifiers, r_i, in Figure 6.12(b)). Indeed, in Figure 6.12(b), r_1 (red), r_2 (blue), r_3 (green),



(a) Cumulative RHODES failures.

(b) Cumulative UNI-CLUB failures.

Figure 6.14: Cumulative localisation failure distributions representing runtime instrumentation resolved according to the *product policy* over the RHODES and UNI-CLUB *product* spaces. In (a), r_1 is the leading value provider, and is required to deal with unsafe drop-outs in localisation of 2 m with a probability of 3.13 % as compared to 4.45 %, 26.39 %, and 15.40 % for r_2, r_3, and r_4, respectively. In contrast, (b) shows r_2 as the leading value provider – required to deal with unsafe drop-outs in localisation of 2 m with a probability of 4.82 % as compared to 12.38 %, 9.95 %, and 5.14 % for r_1, r_3, and r_4, respectively.

and r_4 (orange) contribute imagery with similarities encoded by the heatmap vectorisation (c.f. Equation 6.23 on page 182) of 0.57 ± 0.08, 0.76 ± 0.03, 0.81 ± 0.01, and 0.83 ± 0.00, respectively, such that r_4 is the leading value provider (in terms of FAB-MAP compatibility) over the ORI *product* space.

Figure 6.13 shows the empirical mean and standard deviation (c.f. Equations 6.18 and 6.19 on page 178) of the "price" of performing a data exchange averaged for all (remaining) team members which identify the agent in question as a responder or *seller*, as computed from the recorded price history, $\mathfrak{M}$ (c.f. Section 6.2.2). From this it is clear that the relationship between agent's assignments in Figure 6.12 is preserved in the beliefs shown in Figure 6.13.

6.4.9 Exploiting Local Expertise

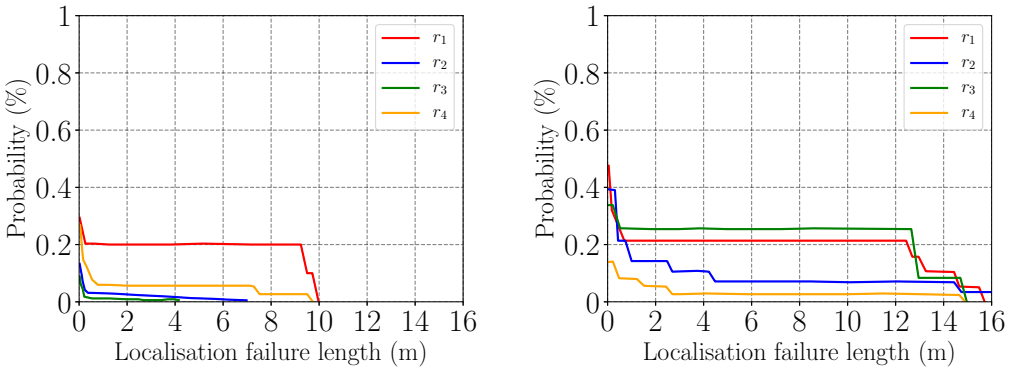
As motivated by "contribution" item 5 on page 7, this experiment presents auxiliary statistics for Section 6.4.8 in order to verify the validity of the bandit game approach to targeting data exchanges which boost localisation outlook while remaining flexible in characterising peers as either dominant or secondary value providers in different regions of the world.

Figures 6.14 and 6.15 thus aggregate statistics for experiments which each involve ten trials in which four agents engage in thirteen forays over a 40 m portion of the ρ_{RHODES}, $\rho_{\text{UNI-CLUB}}$, ρ_{HOLYWELL}, and ρ_{TRINITY} *products* using the *epsilon-greedy sales policy*, a *choice tool* implementation which considers FAB-MAP compatibility between exchanged *patch representations*, as well as restricted *throttling tool* and *ranking tool* implementations (c.f. Equation 5.3 on page 132 and Equation 5.4 on page 133).

The leading value providers are set by assigning datasets logs by visual similarity (c.f. Section 6.3) in the manner shown in Figure 6.12 for each of these *products* separately.

Indeed, in Figure 6.14(a), r_1 (red) has been assigned the dataset logs with the highest similarities encoded by the *heatmap tool*, and achieves an AUC summarising the cumulative failure distributions of 0.16 as compared to 0.39, 1.86, and 0.73 for its peers (r_2, blue, r_3, green, and r_4, orange, respectively). The same effect is observed in Figures 6.14(b), 6.15(a) and 6.15(b), where r_1, r_2, and r_3 take the leading-value provider role for the UNI-CLUB, HOLYWELL and TRINITY *products*, respectively.

Here it is clear that the performance of individual agents must be man-



(a) Cumulative HOLYWELL failures.

(b) Cumulative TRINITY failures.

Figure 6.15: Cumulative localisation failure distributions representing runtime instrumentation resolved according to the *product policy* over the HOLYWELL and TRINITY *product* spaces. In (a), r_3 is the leading value provider, and is required to deal with unsafe drop-outs in localisation of 2 m with a probability of 0.82 % as compared to 17.76 %, 2.47 %, and 5.35 % for r_1, r_2, and r_4, respectively. In contrast, (b) shows r_4 as the leading value provider – required to deal with unsafe drop-outs in localisation of 2 m with a probability of 5.57 % as compared to 22.10 %, 13.79 %, and 26.10 % for r_1, r_2, and r_3, respectively.

aged by a combination of canny selection of content (which is given a value formulated in distinct contexts) and wise engagement with trusted peers (while remaining ready to engage with less trusted peers).

6.5 Summary

This chapter has shown how communities of *canonical DM client* processes which are interfacing through the VCS interpretation of Chapter 3 by means of various strategies for the coordination of the processing blocks described in Chapter 5 can trade-off navigation precision against the resourcing concerns of Chapter 2, in contrast to the idealised scenarios analysed in the

experiments of Section 5.4.

Section 6.1 described how the relative quality of *node representation* data packets is embedded in *throttling tool* and *ranking tool* implementations and used by *buyers* and *sellers* to both reduce the processing, storage, and communication loads as well as align the divergent *patch representations* presented to the *commutation policy* for improved localisation outlook and compatibility in versioning activities.

Sections 6.1.1 and 6.1.2 applied these insights to the centralised trading configuration (c.f. Section 5.1.1) and coordination of roles (c.f. Section 5.2.2), where the idealised whole-database deployment experimented with in Section 5.4.3 was approximated by correlating temporal and "Path Memory" statistics within the privileged *graph representation* before deployment and shown in Section 6.4.1 to reduce the disk and CPU loads while maintaining acceptable localisation performance.

In lieu of a privileged *graph representation* for centralised correlation of this kind, however, Section 6.1.3 applied relative (i.e. FAB-MAP compatibility) and absolute (i.e. quality of ego-motion estimation) metrics on *node representation* quality through the *choice tool* as already used in versioning by the *commutation policy*. Here, the resource impacts of the decentralised trading configuration (c.f. Section 5.1.2) and coordination of tasks (c.f. Section 5.2.3) was shown through the case-studies of Section 6.4.3 to be manageable, with some departure from the idealised boosts to localisation performance in Section 5.4.4 analysed experimentally in Section 6.4.2.

Section 6.4.6 verified that FAB-MAP compatibility lends itself well to optimising the number of matches discovered in repeated branching model

manipulations and calls to the *commutation policy*. Section 6.1.4 considered this compatibility within regions of the world as demarcated by the *product policy*, where Sections 6.4.4 and 6.4.5 showed how this resolved *product* space is beneficial to systems limited by processing power or bandwidth as well as how it should best be explored by requesting agents through a *shopping policy*, respectively.

Section 6.2.1 characterised the measurements inherent to the *ranking tool* and *throttling tool* as a normalised cost-reward ratio (or "price") for completed data exchanges, the historical record of which – as recorded by the *belief tool* – was defined in Section 6.2.2, and used to inform the decisions made by the *sales policy* by casting provider selection as a stochastic scheduling problem around the anticipated regret in executing a data exchange with another identifiable member of the MAS. Section 6.4.7 verified experimentally that this bandit game approach can be used to tune the pairwise compatibility of exchanged *patch representations* as well as approximate the idealised localisation outlook achieved by unrestricted, comprehensive exchanges with the entire team.

Finally, Section 6.3 matured the adjacent DM simulation framework (c.f. *skeleton tool*, Section 4.4.3 and *flow representation*, Section 5.3.1) by discussing a *heatmap tool* which was applied to the case-studies of Sections 6.4.8 and 6.4.9 to verify that the flexibility of this chapter's dual approach of targeting content and provider by means of instrumentation at the resolution of the itemised data exchange protocol of Section 3.4.2 allows DM agents to recover a notion of leading-value provider in distinct parts of the world and adjust their versioning activities accordingly.

Chapter 7

Conclusion

This chapter reemphasises the motivation for this thesis and summarises the major ideas, results, and contributions presented in Chapters 2 to 6, before finishing with a discussion on the future directions of this work, as well as some closing remarks.

7.1 Summary

This thesis culminated in a totally decentralised framework for the targeted and nuanced exchange of navigation expertise amongst *fleet*s of arbitrarily configured robots in a highly variable world.

Developing the Paradigm We developed the representations that mapping DM clients must conform to in Chapter 3, which abstracted localisation as a metric graph relationship, through the lens of a *graph representation*. We showed how this *graph representation* could be manipulated under the

action of a *patch representation* in unordered sets, and made some hard requirements on the versioning techniques that DM clients should employ.

Exploring the Paradigm We showed through examples in Chapters 2 to 4 how a variety of experience mappers could conform to these representations, and instantiated a particular EBN instance, which became the *canonical DM client* which we used immediately to place even stricter requirements on the *commutation policy* in terms of the coherence of data across the *fleet*, as well as the integrity in representation for individual agents.

Centralising the Paradigm We showed that a perspective of this framework as a *central authority* is capable of creating maps in a fraction of the time required by single-agent EBN, with no significant loss in localisation robustness, and could leverage experience within the *fleet* for more robust localisation of novice agents, while being able to furnish robots on real-world forays with maps which required much less storage.

Distributing the Paradigm Indeed, fettering the *canonical DM client* to the versioning operations was shown in Section 4.5.4 to aid localisation performance as well as expedite the discovery of related database content in pairwise interactions. In Section 4.5.3 we delved somewhat into the instrumentation supporting these versioning operations, and showed how the *canonical DM client*, as expressed through the *commutation policy*, reached an acceptable steady state in database corruption. In Section 5.4.6 we showed a selection of trading partner made for more lenient demands on field robots in terms of the available CPU, disk, and network resources. Indeed, in Sec-

tion 5.4.5 we showed how these requirements scaled gracefully with the size of the *fleet* that was engaging with the DM.

Managing the Paradigm Further to this, in Section 6.4.2 we showed how assigning value to content flowing through the DM could significantly alleviate resource constraints, for an acceptable drop in performance. Expanding the value concept to peer interactions, we showed in Section 6.4.7 that a mixture of exploiting the most trusted trading partner with an occasional exploration of the value of unknown or less comprehensively characterised trading partners was superior to pure exploitation. Here, Section 6.4.9 provided more detail, where we showed how one member of the *team*, which was consistently providing more value to the DM, emerged as the most trusted *vendor* in the *Data Market*. Finally, in Section 6.4.4, we showed how these trust factors could be distributed through the *fleet* and leveraged to dictate a more canny querying of content in the form of an unsupervised, mission-dependent "shopping list" which targeted localisation excellence. We show that, by refining market policies over time, agents achieve improved localisation in a directed and accelerated manner.

7.2 Future Work

While this thesis has investigated the boons that an informed and discriminative approach to trading with peers can make available to vision-only localisation performance, future work will focus on applying these concepts to adjacent but related problems, where these systems are notorious in their

susceptibility to design choices and offer a resistance to generalisation.

Plug-and-play pose source generalisation The navigation frameworks investigated [Linegar et al., 2015, Gadd and Newman, 2015] in this work employ vaguely distinct approaches to representing the topology of the world, and significantly distinct characterisations of place as stored on database nodes (expressed through the *node representation*). This software engineering work will entail abstracting the source of pose to conform to a standard data format, by: (a) Moving the estimation into a general purpose pose estimation toolbox, and (b) Interfacing with a canonical database format (or *graph representation*). Additionally, alternative *commutation policies* may employ the recent work in [Giamou et al., 2017] which takes into account both the total amount of exchanged data and the induced division of labor between the participating robots.

A swathe of weak localisers as *product policies* In this work we have relied upon a vast set of labels and GPS traces for a publicly available dataset [Maddern et al., 2017] to demarcate the regions of the world that each robot contributes expertise to, and appraises as a commodity, or *product*, for purchase. Here, [Cieslewski and Scaramuzza, 2017] have assigned responsibility for different words in a visual vocabulary to members of a *team* of robots, which prove to be an effective way for robots to both gain a more nuanced understanding of the *product* range in terms of its appearance, as well as a strong mechanism for assigning local expertise to individual robots.

Abstract expertise and value There is no reason why the fundamental unit of exchange in the DM (which in this thesis take the form database nodes – *node representations* – and their connectivity – *edge representations* – as an expression of a the local neighbourhood around a robot's current pose) cannot be substituted for a general set of time series metrics which convey experience and expertise. Additionally, there is no reason to limit the exploitation and recommendation strategies to autonomous vehicles, where passive exchange between connected processes will in general prove to be beneficial too. Here, ongoing work includes modelling risk and dynamic authorisation over a *fleet* of insured vehicles to optimise overall autonomous *fleet* safety and operation.

Demonstrations in the public domain The next two years should see some of the techniques presented in this thesis culminate in six co-operative autonomous vehicles performing mixed urban and motorway driving routes in a live-traffic environment between Oxford and London.

7.3 Closing Remark

This thesis has presented the DM, a set of protocols and policies for the beneficial exchange of navigation expertise between agents which are localising in a variable world. The key message here is that agents can "experience" and represent the world in a multitude of ways, which is not isolated from similar knowledge that their peers have accumulated. Leveraging referrals, however, should be done with discrimination and intent, and we trust that

this work has contributed a set of guiding principles which will accelerate the materialisation of *fleet*s of connected, mutually assistive, autonomous vehicles in the real-world.

Appendix A

Datasets

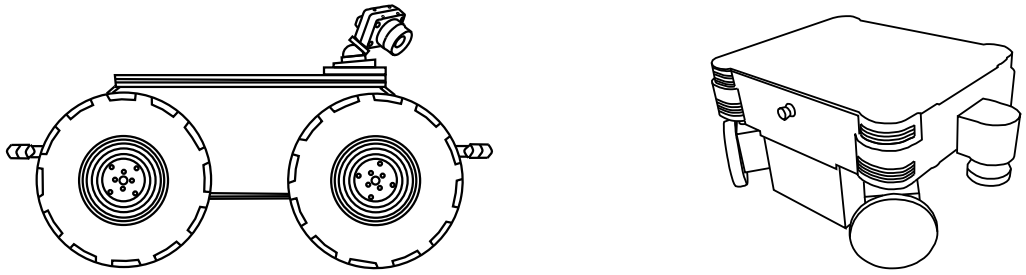
This appendix details the platforms, sensors, and routes that were used in the motivating examples and results presented in Chapters 2 to 6.

A.1 Textures

The motivating examples from Chapters 2 to 4 were drawn from early work towards this thesis on the EBN paradigm [Gadd and Newman, 2015], in which downwards-facing cameras mounted on a variety of small robots were used to track the perceived texture to the front of the platform's wheels.

A.1.1 Warehouse

The examples in Chapters 2 to 4 were courtesy two mobile robotic platforms particularly suited to warehousing.



(a) *Clearpath Husky A200* and *Pointgrey Firefly*.

(b) *Neobotix MP-70*.

Figure A.1: Wireframe depictions of *Clearpath Husky A200*, *Pointgrey Firefly*, and *Neobotix MP-70*. As discussed in Section 2.1.2, these platforms exhibit significantly different motion on manual or automatic control, as the *Neobotix MP-70* has approximately holonomic control thanks to double-actuated caster wheels.

Platform

A *Clearpath Husky A200* and a *Neobotix MP-70*, shown in Figures A.1(a) and A.1(b) respectively.

Hardware

Offline examples were obtained on a machine with 2.3 GHz *Intel Core i7* processor and 1600 MHz DDR 8 GiB RAM.

Imagery

Images were captured by a *Pointgrey Firefly*, depicted mounted on the *Clearpath Husky A200* in Figure A.1(a), at 60 Hz and a resolution of 752 px×480 px, and present us with textures such as:

1. A highly textured and well-lit office floor surface, Figure A.2(a), and



(a) Office. (b) Warehouse. (c) Sidewalk.

Figure A.2: Texture samples.

2. A dramatically lit or rather featureless Earth picking and shelving area, Figure A.2(b), and

3. Irregular and cluttered sidewalks, Figure A.2(c).

A.1.2 Martian analogue

The exemplar experience mapper shown in Sections 2 to 4 – particularly in Example 2.2.2 – was tested during *Mars Utah Rover Field Investivation* (MURFI), a planetary analogue mission run by the *United Kingdom Space Agency* (UKSA) in collaboration with the *Canadian Space Agency* (CSA) between 22nd October and 13th November 2016.

Platform

The rover platform comprised a ARCQ14 robot from *Advanced Robotics Concepts* (ARC), as shown in Figure A.3.

Location

This ARCQ14 platform was repeatedly driven around a highly textured and somewhat flat area near the MURFI base, a field site near *Hanksville* (*Utah*,



Figure A.3: ARCQ14 and *Point Grey Grasshopper* mounted towards the front of the rover platform bed.

USA).

Extent

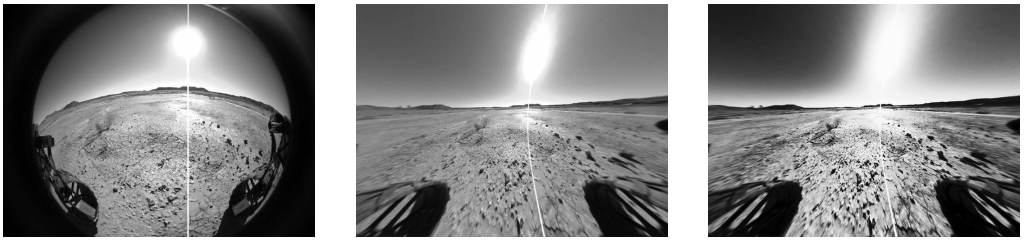
The route was piloted at regular intervals (eight traverses) over a period of twenty-four hours, totalling approximately 800 m.

Imagery

As shown in Figure A.4(a), the greyscale imagery was captured by a *Point Grey Grasshopper* monocular camera at resolution 1036 px×1084 px. As such, the data presented us with variation in environmental appearance, predominantly due to the waxing and waning of available light as the sun changed position.

A.2 Urban

The results from Chapters 4 to 6 are representative of urban scenes, and detail the entry of a passenger vehicle as an experience mapper into the DM.



(a) Greyscale imagery captured by the fish-eye camera.

(b) Undistorted by applying the camera intrinsics.

(c) Histogram equalisation can remove specularities caused by harsh glare.

Figure A.4: Sample imagery from the downward-facing *Point Grey Grasshopper* mounted on ARCQ14, as well as examples of simple preprocessing steps taken to ensure good estimation and some robustness to changes in lighting.

A.2.1 Begbroke

This dataset was used to generate the results presented in Chapter 4, and we refer to this set of logs as BEGBROKE-24HR.

Platform

A *Nissan Leaf RobotCar*, a schematic for which is shown in Figure A.6.

Location

BEGBROKE-24HR was collected in Summer 2013 by driving around Begbroke Science Park in North Oxford, GPS traces for which are shown in Figure A.5(b).

Extent

The 700 m loop was driven at regular intervals and at all times of day and night between August 6th and September 5th (30 d) in 2013, totalling 100 km.



(a) Begbroke Science Park, North Oxford sample stereo imagery.

(b) Begbroke Science Park, North Oxford GPS traces.

Figure A.5: The BEGBROKE-24HR dataset.

Imagery

The stereo images were obtained from a *Point Grey Bumblebee2* stereo camera at resolution 384 px×512 px, as shown in Figure A.5(a). It presents us with significant challenges in appearance change due to atmospheric (from glaring sun to gloomy overcast), lighting (at all times of day and night) and structural (fluctuating arrangements of traffic) effects.

A.2.2 Oxford

This dataset was used towards the experiments presented in Chapters 5 and 6, and we refer to this set of logs as the *Oxford RobotCar Dataset* [Maddern et al., 2017].

Platform

The *Nissan Leaf RobotCar*, a schematic for which is shown in Figure A.6.

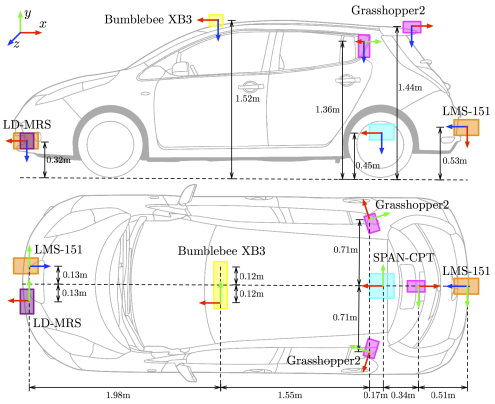


Figure A.6: Top-down and side-view schematics for *Nissan Leaf RobotCar* complete with sensor suite and frames of reference, taken from the *Oxford RobotCar Dataset* [Maddern et al., 2017] and available at `robotcar-dataset.robots.ox.ac.uk`.

Hardware

We consider the behaviour of arbitrarily sized *fleet*s of agents which run as processes on hardware (limited to four cores and 16 GiB RAM) that share no physical medium except for a network connection (with latencies ranging from 50 to 500 ms).

Location

Figure A.7 shows GPS traces for a typical foray over the full extent of the *Oxford RobotCar Dataset*, where the colour palette is expressive of a candidate *product* space.

Extent

Table A.1 shows an exemplar demarcation as a portion of the 55 logs, or 446 km, of which has been labelled and categorised in this work, as discussed in Section 3.4. Each catalogue consists of a range of *products* which corre-

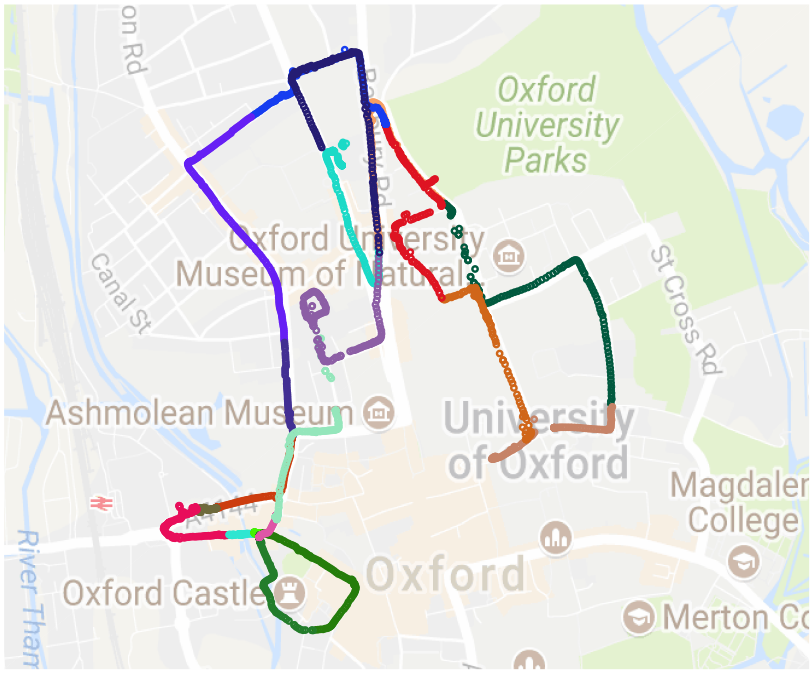


Figure A.7: GPS traces and an example *product* space over the *Oxford RobotCar Dataset*.

Catalogue	Category	Products	Foray Length
ST-ANNES	college	12	143
BEVINGTON	street	15	180
RHODES	house	21	264
TRINITY	college	28	384
BLACKHALL	street	17	210
OBSERVATORY	street	26	322
ORI	lab	36	450
BROAD	street	18	440
MATERIALS	street	24	300
ROYAL-OAK	pub	32	330
UNI-CLUB	field	42	430
HOLYWELL	street	18	181

Table A.1: Exemplar demarcation of the *Oxford RobotCar Dataset*.

spond to drivable sections of the world in the vicinity of the street or local attraction. Additionally, each categorised catalogue is further resolved into reasonably sized sections which *vendors* treat as *products* for appraisal and purchase.

Imagery

Figure A.8 shows typical scenes from four of the *product* categories mentioned in Table A.1.



Figure A.8: Typical appearance change within BEVINGTON, BROAD, BLACKHALL, and OBSERVATORY *product* categories.

Bibliography

[Alahi et al., 2012] Alahi, A., Ortiz, R., and Vandergheynst, P. (2012). FREAK: Fast Retina Keypoint. In *Computer vision and pattern recognition (CVPR), 2012 IEEE conference on*, pages 510–517. IEEE.

[Angiuli et al., 2014] Angiuli, C., Morehouse, E., Licata, D. R., and Harper, R. (2014). Homotopical patch theory. In *ACM SIGPLAN Notices*, volume 49, pages 243–256. ACM.

[Auer et al., 2002] Auer, P., Cesa-Bianchi, N., Freund, Y., and Schapire, R. E. (2002). The nonstochastic multiarmed bandit problem. *SIAM Journal on Computing*, 32(1):48–77.

[Bahr et al., 2009] Bahr, A., Walter, M. R., and Leonard, J. J. (2009). Consistent cooperative localization. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3415–3422. IEEE.

[Bailey et al., 2011] Bailey, T., Bryson, M., Mu, H., Vial, J., McCalman, L., and Durrant-Whyte, H. (2011). Decentralised cooperative localisation for heterogeneous teams of mobile robots. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 2859–2865. IEEE.

[Bailey and Durrant-Whyte, 2006] Bailey, T. and Durrant-Whyte, H. (2006). Simultaneous localization and mapping (slam): Part ii. *IEEE Robotics & Automation Magazine*, 13(3):108–117.

[Balme et al., 2017] Balme, M. R. et al. (2017). UK Space Agency "Mars Utah Rover Field Investigation 2016" (MURFI 2016): Overview of mission, aims and progress. In *48th Lunar and Planetary Science Conference*.

[Bay et al., 2006] Bay, H., Tuytelaars, T., and Van Gool, L. (2006). SURF: Speeded Up Robust Features. In *European Conference on Computer Vision (ECCV)*, pages 404–417. Springer.

[Biber et al., 2005] Biber, P., Duckett, T., et al. (2005). Dynamic maps for long-term operation of mobile service robots. In *Robotics: science and systems*, pages 17–24.

[Blanc et al., 2005] Blanc, G., Mezouar, Y., and Martinet, P. (2005). Indoor navigation of a wheeled mobile robot along visual routes. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3354–3359. IEEE.

[Bosik and Uyar, 1991] Bosik, B. S. and Uyar, M. Ü. (1991). Finite state machine based formal methods in protocol conformance testing: from theory to implementation. *Computer Networks and ISDN Systems*, 22(1):7–33.

[Buczko and Willert, 2016] Buczko, M. and Willert, V. (2016). How to distinguish inliers from outliers in visual odometry for high-speed automotive applications. In *Intelligent Vehicles Symposium (IV), 2016 IEEE*, pages 478–483. IEEE.

[Bundy and Wallen, 1984] Bundy, A. and Wallen, L. (1984). Breadth-first search. In *Catalogue of Artificial Intelligence Tools*, pages 13–13. Springer.

[Cai et al., 2011] Cai, G., Chen, B. M., and Lee, T. H. (2011). *Unmanned rotorcraft systems*. Springer Science & Business Media.

[Calonder et al., 2010] Calonder, M., Lepetit, V., Strecha, C., and Fua, P. (2010). BRIEF: Binary Robust Independent Elementary Features. *European Conference on Computer Vision (ECCV)*, pages 778–792.

[Cardwell et al., 2000] Cardwell, N., Savage, S., and Anderson, T. (2000). Modeling TCP latency. In *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 3, pages 1742–1751. IEEE.

[Carrillo-Arce et al., 2013] Carrillo-Arce, L. C., Nerurkar, E. D., Gordillo, J. L., and Roumeliotis, S. I. (2013). Decentralized multi-robot cooperative localization using covariance intersection. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1412–1417. IEEE.

[Chow and Liu, 1968] Chow, C. and Liu, C. (1968). Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, 14(3):462–467.

[Churchill, 2012] Churchill, W. (2012). *Experience based navigation: Theory, practice and implementation*. PhD thesis, University of Oxford.

[Churchill and Newman, 2012a] Churchill, W. and Newman, P. (2012a). Continually improving large scale long term visual navigation of a vehicle in dynamic urban environments. In *Intelligent Transportation Systems (ITSC), 2012 15th International IEEE Conference on*, pages 1371–1376. IEEE.

[Churchill and Newman, 2012b] Churchill, W. and Newman, P. (2012b). Practice makes perfect? managing and leveraging visual experiences for lifelong navigation. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 4525–4532. IEEE.

[Churchill and Newman, 2013] Churchill, W. and Newman, P. (2013). Experience-based navigation for long-term localisation. *The International Journal of Robotics Research (IJRR)*, 32(14):1645–1661.

[Cieslewski et al., 2015] Cieslewski, T., Lynen, S., Dymczyk, M., Magnenat, S., and Siegwart, R. (2015). Map API – Scalable decentralized map building for robots. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 6241–6247. IEEE.

[Cieslewski and Scaramuzza, 2017] Cieslewski, T. and Scaramuzza, D. (2017). Efficient decentralized visual place recognition using a distributed inverted index. *IEEE Robotics and Automation Letters*, 2(2):640–647.

[Comer, 1988] Comer, D. (1988). *Internetworking with TCP/IP: Principles, protocols, and architecture*. Prentice Hall, Englewood Cliffs, NJ.

[Connolly and Begg, 2005] Connolly, T. M. and Begg, C. E. (2005). *Database systems: a practical approach to design, implementation, and management*. Pearson Education.

[Corke et al., 2013] Corke, P., Paul, R., Churchill, W., and Newman, P. (2013). Dealing with shadows: Capturing intrinsic scene appearance for image-based outdoor localisation. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 2085–2092. IEEE.

[Cummins and Newman, 2008] Cummins, M. and Newman, P. (2008). FAB-MAP: Probabilistic localization and mapping in the space of appearance. *The International Journal of Robotics Research (IJRR)*, 27(6):647–665.

[Cummins and Newman, 2011] Cummins, M. and Newman, P. (2011). Appearance-only SLAM at large scale with FAB-MAP2.0. *The International Journal of Robotics Research (IJRR)*, 30(9):1100–1123.

[Cunningham et al., 2013] Cunningham, A., Indelman, V., and Dellaert, F. (2013). DDF-SAM 2.0: Consistent distributed smoothing and mapping. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 5220–5227. IEEE.

[Cunningham et al., 2012] Cunningham, A., Wurm, K. M., Burgard, W., and Dellaert, F. (2012). Fully distributed scalable smoothing and mapping with robust multi-robot data association. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 1093–1100. IEEE.

[Dagit, 2009] Dagit, J. (2009). *Type-correct changes – A safe approach to version control implementation*. PhD thesis.

[Dequaire et al., 2016] Dequaire, J., Tong, C. H., Churchill, W., and Posner, I. (2016). Off the beaten track: Predicting localisation performance in visual teach and repeat. In *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, pages 795–800. IEEE.

[Diestel, 2005] Diestel, R. (2005). Graph theory. 2005. *Grad. Texts in Math*, 101.

[Dijkstra, 1959] Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1):269–271.

[Dissanayake et al., 2001] Dissanayake, M. G., Newman, P., Clark, S., Durrant-Whyte, H. F., and Csorba, M. (2001). A solution to the simultaneous localization and map building (slam) problem. *IEEE Transactions on robotics and automation*, 17(3):229–241.

[Dushnik and Miller, 1941] Dushnik, B. and Miller, E. W. (1941). Partially ordered sets. *American Journal of Mathematics*, 63(3):600–610.

[Duygulu et al., 2002] Duygulu, P., Barnard, K., de Freitas, J. F., and Forsyth, D. A. (2002). Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary. In *European conference on computer vision*, pages 97–112. Springer.

[Dymczyk et al., 2015] Dymczyk, M., Lynen, S., Cieslewski, T., Bosse, M., Siegwart, R., and Furgale, P. (2015). The gist of maps – Summarising experiences for life-long localisation. In *Proc. IEEE International Conference on Robotics and Automation (ICRA2015)*.

[Fischler and Bolles, 1981] Fischler, M. A. and Bolles, R. C. (1981). Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395.

[Forster et al., 2013] Forster, C., Lynen, S., Kneip, L., and Scaramuzza, D. (2013). Collaborative monocular slam with multiple micro aerial vehicles. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3962–3970. IEEE.

[Forster et al., 2014] Forster, C., Pizzoli, M., and Scaramuzza, D. (2014). SVO: Fast semi-direct monocular visual odometry. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 15–22. IEEE.

[Fourie et al., 2017] Fourie, D., Claassens, S., Pillai, S., Mata, R., and Leonard, J. (2017). Slamindb: Centralized graph databases for mobile robotics. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 6331–6337. IEEE.

[Fox et al., 2000] Fox, D., Burgard, W., Kruppa, H., and Thrun, S. (2000). A probabilistic approach to collaborative multi-robot localization. 8(3):325–344.

[Fraundorfer and Scaramuzza, 2012] Fraundorfer, F. and Scaramuzza, D. (2012). Visual Odometry: Part II: Matching, Robustness, Optimization, and Applications. *Robotics & Automation Magazine, IEEE*, 19(2):78–90.

[Frias-Martinez et al., 2004] Frias-Martinez, V., Sklar, E., and Parsons, S. (2004). Exploring auction mechanisms for role assignment in teams of autonomous robots. In *Robot Soccer World Cup*, pages 532–539. Springer.

[Furgale and Barfoot, 2010] Furgale, P. and Barfoot, T. D. (2010). Visual teach and repeat for long-range rover autonomy. *Journal of Field Robotics*, 27(5):534–560.

[Furgale et al., 2015] Furgale, P., Pradalier, C., and Barfoot, T. D. (2015). Special issue on calibration for field robotics. *Journal of Field Robotics*, 32(5):629–631.

[Gadd and Newman, 2015] Gadd, M. and Newman, P. (2015). A framework for infrastructure-free warehouse navigation. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Seattle, WA, USA.

[Gadd and Newman, 2016] Gadd, M. and Newman, P. (2016). Checkout my map: Version control for fleetwide visual localisation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Daejeon, South Korea.

[Gadd and Newman, 2018] Gadd, M. and Newman, P. (2018). The Data Market: Policies for Decentralised Visual Localisation. *ArXiv e-prints*.

[Gauss, 1855] Gauss, C. F. (1855). *Méthode des moindres carrés: Mémoires sur la combinaison des observations*. Mallet-Bachelier.

[Giamou et al., 2017] Giamou, M., Khosoussi, K., and How, J. P. (2017). Talk Resource-Efficiently to Me: Optimal Communication Planning for Distributed SLAM Front-Ends. *arXiv preprint arXiv:1709.06675*.

[Glover et al., 2012] Glover, A., Maddern, W., Warren, M., Reid, S., Milford, M., and Wyeth, G. (2012). OpenFABMAP: An open source toolbox for appearance-based loop closure detection. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 4730–4735. IEEE.

[Grisetti et al., 2010] Grisetti, G., Kummerle, R., Stachniss, C., and Burgard, W. (2010). A tutorial on graph-based SLAM. *IEEE Intelligent Transportation Systems Magazine*, 2(4):31–43.

[Hamilton, 1844] Hamilton, W. R. (1844). II. On quaternions; or on a new system of imaginaries in algebra. *Philosophical Magazine Series 3*, 25(163):10–13.

[Hamming, 1950] Hamming, R. W. (1950). Error detecting and error correcting codes. *Bell Labs Technical Journal*, 29(2):147–160.

[Han et al., 2017] Han, F., Yang, X., Deng, Y., Rentschler, M., Yang, D., and Zhang, H. (2017). Sral: Shared representative appearance learning for long-term visual place recognition. *IEEE Robotics and Automation Letters*, 2(2):1172–1179.

[Harary, 2005] Harary, F. (2005). Structural models: An introduction to the theory of directed graphs.

[Harris and Stephens, 1988] Harris, C. and Stephens, M. (1988). A combined corner and edge detector. In *Alvey vision conference*, volume 15, pages 10–5244. Manchester, UK.

[Harrison and Newman, 2011] Harrison, A. and Newman, P. (2011). Ticsync: Knowing when things happened. In *ICRA*, pages 356–363.

[Hartley and Zisserman, 2003] Hartley, R. and Zisserman, A. (2003). *Multiple view geometry in computer vision*. Cambridge University Press.

[Ho and Newman, 2007] Ho, K. L. and Newman, P. (2007). Detecting loop closure with scene sequences. *International Journal of Computer Vision*, 74(3):261–286.

[Huber, 1964] Huber, P. J. (1964). Robust estimation of a location parameter. *The annals of mathematical statistics*, 35(1):73–101.

[Isola et al., 2017] Isola, P., Zhu, J.-Y., Zhou, T., and Efros, A. A. (2017). Image-to-image translation with conditional adversarial networks. *arXiv preprint*.

[Khamsi and Kirk, 2011] Khamsi, M. A. and Kirk, W. A. (2011). *An introduction to metric spaces and fixed point theory*, volume 53. John Wiley & Sons.

[Klein and Murray, 2007] Klein, G. and Murray, D. (2007). Parallel tracking and mapping for small AR workspaces. In *Mixed and Augmented Reality, 2007. ISMAR 2007. 6th IEEE and ACM International Symposium on*, pages 225–234. IEEE.

[Knuth, 2007] Knuth, D. E. (2007). Seminumerical algorithms.

[Konolige and Bowman, 2009] Konolige, K. and Bowman, J. (2009). Towards lifelong visual maps. In *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, pages 1156–1163. IEEE.

[Kosaraju, 1973] Kosaraju, S. R. (1973). Limitations of Dijkstra's semaphore primitives and petri nets. *ACM SIGOPS Operating Systems Review*, 7(4):122–126.

[Krajník et al., 2017] Krajník, T., Cristóforis, P., Kusumam, K., Neubert, P., and Duckett, T. (2017). Image features for visual teach-and-repeat navigation in changing environments. *Robotics and Autonomous Systems*, 88:127–141.

[Krajník et al., 2016] Krajník, T., Fentanes, J. P., Hanheide, M., and Duckett, T. (2016). Persistent localization and life-long mapping in changing environments using the frequency map enhancement. In *Intelligent Robots*

and Systems (IROS), 2016 IEEE/RSJ International Conference on, pages 4558–4563. IEEE.

[Krajnik et al., 2017] Krajnik, T., Majer, F., Halodová, L., Bayer, J., Vintr, T., and Faigl, J. (2017). Navigation without localisation: reliable teach and repeat based on the convergence theorem. *arXiv preprint arXiv:1711.05348*.

[Labbé and Michaud, 2018] Labbé, M. and Michaud, F. (2018). Long-term online multi-session graph-based splam with memory management. *Autonomous Robots*, 42(6):1133–1150.

[Langford and Zhang, 2008] Langford, J. and Zhang, T. (2008). The epoch-greedy algorithm for multi-armed bandits with side information. In *Advances in neural information processing systems*, pages 817–824.

[Leach et al., 2005] Leach, P. J., Mealling, M., and Salz, R. (2005). A universally unique identifier (UUID) URN namespace. Technical report.

[Leutenegger et al., 2011] Leutenegger, S., Chli, M., and Siegwart, R. Y. (2011). BRISK: Binary robust invariant scalable keypoints. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2548–2555. IEEE.

[Levenberg, 1944] Levenberg, K. (1944). A method for the solution of certain non-linear problems in least squares. *Quarterly of Applied Mathematics*, 2(2):164–168.

[Linegar, 2016] Linegar, C. (2016). *Vision-Only Localisation Under Extreme Appearance Change*. PhD thesis, University of Oxford, Oxford, United Kingdom.

[Linegar et al., 2015] Linegar, C., Churchill, W., and Newman, P. (2015). Work Smart, Not Hard: Recalling Relevant Experiences for Vast-Scale but Time-Constrained Localisation. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Seattle, WA, USA.

[Loeliger and McCullough, 2012] Loeliger, J. and McCullough, M. (2012). *Version Control with Git: Powerful tools and techniques for collaborative software development*. "O'Reilly Media, Inc.".

[Lowe, 2004] Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110.

[Ma, 2004] Ma, Y. (2004). *An invitation to 3-d vision: From images to geometric models*, volume 26. Springer.

[MacTavish et al., 2017] MacTavish, K., Paton, M., and Barfoot, T. D. (2017). Visual triage: A bag-of-words experience selector for long-term visual route following. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 2065–2072. IEEE.

[Maddern et al., 2017] Maddern, W., Pascoe, G., Linegar, C., and Newman, P. (2017). 1 Year, 1000km: The Oxford RobotCar Dataset. *The International Journal of Robotics Research (IJRR)*, 36(1):3–15.

[Maddern et al., 2015] Maddern, W., Pascoe, G., and Newman, P. (2015). Leveraging Experience for Large-Scale LIDAR Localisation in Changing Cities. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Seattle, WA, USA.

[Marquardt, 1963] Marquardt, D. W. (1963). An algorithm for least-squares estimation of nonlinear parameters. *Journal of the Society for Industrial and Applied Mathematics*, 11(2):431–441.

[McAfee and McMillan, 1987] McAfee, R. P. and McMillan, J. (1987). Auctions and bidding. *Journal of economic literature*, 25(2):699–738.

[McManus et al., 2014] McManus, C., Churchill, W., Maddern, W., Stewart, A. D., and Newman, P. (2014). Shady dealings: Robust, long-term visual localisation using illumination invariance. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 901–906. IEEE.

[Milford and Wyeth, 2010] Milford, M. and Wyeth, G. (2010). Persistent navigation and mapping using a biologically inspired SLAM system. *The International Journal of Robotics Research (IJRR)*, 29(9):1131–1153.

[Milford and Wyeth, 2012] Milford, M. J. and Wyeth, G. F. (2012). SeqSLAM: Visual route-based navigation for sunny summer days and stormy winter nights. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 1643–1649. IEEE.

[Mühlfellner et al., 2016] Mühlfellner, P., Bürki, M., Bosse, M., Derendarz, W., Philippsen, R., and Furgale, P. (2016). Summary maps for lifelong visual localization. *Journal of Field Robotics*, 33(5):561–590.

[Muja and Lowe, 2009] Muja, M. and Lowe, D. G. (2009). Flann, fast library for approximate nearest neighbors. In *International Conference on Computer Vision Theory and Applications (VISAPP09)*, volume 3. INSTICC Press.

[Nagel, 2005] Nagel, W. (2005). *Subversion Version Control: Using the Subversion Version Control System in Development Projects*. Prentice Hall PTR.

[Naseer et al., 2014] Naseer, T., Spinello, L., Burgard, W., and Stachniss, C. (2014). Robust visual robot localization across seasons using network flows. In *AAAI*, pages 2564–2570.

[Nelson et al., 2015] Nelson, P., Linegar, C., and Newman, P. (2015). Building, Curating, and Querying Large-scale Data Repositories for Field Robotics Applications. In *International Conference on Field and Service Robotics (FSR)*, Toronto, ON, Canada.

[Neubert et al., 2013] Neubert, P., Sunderhauf, N., and Protzel, P. (2013). Appearance change prediction for long-term navigation across seasons. In *Mobile Robots (ECMR), 2013 European Conference on*, pages 198–203. IEEE.

[Newman, 2008] Newman, P. M. (2008). MOOS – Mission orientated operating suite. Technical report.

[Nistér et al., 2004] Nistér, D., Naroditsky, O., and Bergen, J. (2004). Visual Odometry. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 1, pages I–652. IEEE.

[Owens and Allen, 2010] Owens, M. and Allen, G. (2010). *The Definitive Guide to SQLite*. Springer.

[Paul et al., 2012] Paul, R., Rus, D., and Newman, P. (2012). How was your day? Online visual workspace summaries using incremental clustering in topic space. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 4058–4065. IEEE.

[Pepperell et al., 2015] Pepperell, E., Corke, P., and Milford, M. (2015). Automatic image scaling for place recognition in changing environments. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*.

[Pepperell et al., 2014] Pepperell, E., Corke, P. I., and Milford, M. J. (2014). All-environment visual place recognition with SMART. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 1612–1618. IEEE.

[Porav et al., 2018] Porav, H., Maddern, W., and Newman, P. (2018). Adversarial training for adverse conditions: Robust metric localisation using appearance transfer. *arXiv preprint arXiv:1803.03341*.

[Riazuelo et al., 2014] Riazuelo, L., Civera, J., and Montiel, J. (2014). C2TAM: A cloud framework for cooperative tracking and mapping. *Robotics and Autonomous Systems*, 62(4):401–413.

[Robert, 2004] Robert, C. P. (2004). *Monte carlo methods*. Wiley Online Library.

[Rosen et al., 2016] Rosen, D., Mason, J., and Leonard, J. (2016). Towards lifelong feature-based mapping in semi-static environments. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 1063–1070, Stockholm, Sweden.

[Rosten and Drummond, 2006] Rosten, E. and Drummond, T. (2006). Machine learning for high-speed corner detection. *European Conference on Computer Vision (ECCV)*, pages 430–443.

[Roundy, 2005] Roundy, D. (2005). Darcs: Distributed version management in Haskell. In *Proceedings of the 2005 ACM SIGPLAN workshop on Haskell*, pages 1–4. ACM.

[Rublee et al., 2011] Rublee, E., Rabaud, V., Konolige, K., and Bradski, G. (2011). ORB: An efficient alternative to SIFT or SURF. In *Computer Vision (ICCV), 2011 IEEE international conference on*, pages 2564–2571. IEEE.

[Scaramuzza et al., 2006] Scaramuzza, D., Martinelli, A., and Siegwart, R. (2006). A toolbox for easily calibrating omnidirectional cameras. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 5695–5701. IEEE.

[Scott et al., 2016] Scott, T., Morye, A. A., Piniés, P., Paz, L. M., Posner, I., and Newman, P. (2016). Choosing a time and place for calibration of lidar-camera systems. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 4349–4356. IEEE.

[Seitz and Kim, 2002] Seitz, S. M. and Kim, J. (2002). The space of all stereo images. *International Journal of Computer Vision*, 48(1):21–38.

[Sibley et al., 2010a] Sibley, G., Mei, C., Reid, I., and Newman, P. (2010a). Planes, Trains and Automobiles – Autonomy for the Modern Robot. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 285–292. IEEE.

[Sibley et al., 2010b] Sibley, G., Mei, C., Reid, I., and Newman, P. (2010b). Vast-scale outdoor navigation using adaptive relative bundle adjustment. *The International Journal of Robotics Research (IJRR)*, 29(8):958–980.

[Sidorov et al., 2014] Sidorov, G., Gelbukh, A., Gómez-Adorno, H., and Pinto, D. (2014). Soft similarity and soft cosine measure: Similarity of features in vector space model. *Computación y Sistemas*, 18(3):491–504.

[Sivic and Zisserman, 2003] Sivic, J. and Zisserman, A. (2003). Video Google: A text retrieval approach to object matching in videos. In *International Conference on Computer Vision*, page 1470. IEEE.

[Smith and Brady, 1997] Smith, S. M. and Brady, J. M. (1997). SUSAN – A new approach to low level image processing. *International Journal of Computer Vision*, 23(1):45–78.

[Sofman et al., 2010] Sofman, B., Bagnell, J. A., and Stentz, A. (2010). Bandit-based online candidate selection for adjustable autonomy. In *Field and Service Robotics*, pages 239–248. Springer.

[Sparck Jones, 1972] Sparck Jones, K. (1972). A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, 28(1):11–21.

[Stenning et al., 2013] Stenning, B. E., McManus, C., and Barfoot, T. D. (2013). Planning using a network of reusable paths: A physical embodiment of a rapidly exploring random tree. *Journal of Field Robotics*, 30(6):916–950.

[Stonebraker and Rowe, 1986] Stonebraker, M. and Rowe, L. A. (1986). *The design of Postgres*, volume 15. ACM.

[Stormo et al., 1982] Stormo, G. D., Schneider, T. D., Gold, L., and Ehrenfeucht, A. (1982). Use of the 'perceptron' algorithm to distinguish translational initiation sites in e. coli. *Nucleic Acids Research*, 10(9):2997–3011.

[Stuelpnagel, 1964] Stuelpnagel, J. (1964). On the parametrization of the three-dimensional rotation group. *SIAM review*, 6(4):422–430.

[Sumaray and Makki, 2012] Sumaray, A. and Makki, S. K. (2012). A comparison of data serialization formats for optimal efficiency on a mobile platform. In *Proceedings of the 6th international conference on ubiquitous information management and communication*, page 48. ACM.

[Thompson, 1999] Thompson, S. (1999). *Haskell: The craft of functional programming*, volume 3. Addison Wesley Reading.

[Verhulst, 2009] Verhulst, P.-F. (2009). Notice sur la loi que la population poursuit dans son accroissement. correspondance mathématique et physique 10: 113–121. Technical report, Retrieved 09/08.

[Wang et al., 2016] Wang, L., Liu, M., and Meng, M. Q.-H. (2016). A pricing mechanism for task oriented resource allocation in cloud robotics. In *Robots and Sensor Clouds*, pages 3–31. Springer.

[Wingerd, 2005] Wingerd, L. (2005). *Practical Perforce*. "O'Reilly Media, Inc.".

[Yeomans et al., 2017] Yeomans, B., Porav, H., Gadd, M., Barnes, D., Dequaire, J., Wilcox, T., Kyberd, S., Venn, S., and Newman, P. (2017). MURFI 2016 - From Cars to Mars: Applying autonomous vehicle navigation methods to a space rover mission. In *Proceedings of the 14th Symposium on Advanced Space Technologies in Robotics and Automation (ASTRA)*, Leiden, Netherlands.

[Zhang and Kleeman, 2009] Zhang, A. M. and Kleeman, L. (2009). Robust appearance based visual route following for navigation in large-scale outdoor environments. *The International Journal of Robotics Research (IJRR)*, 28(3):331–356.