Large-scale Multi-robot Mapping in MAGIC 2010

Robert Reid, *Member, IEEE,* Thomas Bräunl, *Senior Member, IEEE*

***Abstract*—We describe a large-scale decentralised multi-robot mapping system that outputs globally optimised metric maps in real-time. The mapping system was used by team WAMbot in the finals of the Multi-Autonomous Ground-robotics International Challenge (MAGIC 2010). Research contributions include a novel large-scale multi-robot graph-based non-linear map optimisation approach, a hybrid decentralised and distributed mapping system and novel graphics processing unit (GPU) based approaches for accelerating intensive map matching and fusion operations. Our mapping system scales linearly with map size and on commodity hardware can easily map a 500m×500m urban area. We demonstrate robust, highly efficient and accurate mapping results from two different fleets of mobile robots. Videos, maps and timing results from the MAGIC 2010 challenge are presented.**

***Index Terms*—Large-scale multi-robot mapping, simultaneous localisation & mapping (SLAM), graph-based SLAM, distributed, decentralised, GPU map fusion, GPU map correlation.**

I. Introduction

THE goal of the Multi-Autonomous Ground-robotics International Challenge (MAGIC 2010) was to design a fleet of unmanned ground vehicles (UGVs) that were capable of exploring and mapping a large (500m×500m) urban area while identifying objects of interest (OOIs). The ultimate objective of the USD $1.2 million challenge was to demonstrate high levels of autonomous control over large groups of UGVs. To robustly map large areas of unknown terrain with teams of heterogeneous UGVs an advanced multi-robot simultaneous localisation and mapping (SLAM) system is required.

SLAM describes the tightly coupled problem of accurately localising a robot within a coordinate frame, while building a map of its unknown environment [1], [2]. In outdoor environments the problem becomes difficult in the presence of noisy sensor data, where wheel slippage and civilian-grade GPS produce distorted position data. Here also random and systematic errors frequently occur in scanning laser range finder (LIDAR) data due to complex beam dynamics in an irregular environment. The SLAM problem has been understood for over a decade and convincing systems for mobile robots have been described in both laboratory and real-world conditions. However, there are limited examples of *robust* SLAM solutions fusing data from *multiple* UGVs into a single globally metric map. The move from single to multiple independent mobile robots introduces considerable complexity [3], [4], [5]. The most significant problems are accurately localising and synchronising multiple robots in a common coordinate system, and fusing multiple streams of mapping data in real-time into a single coherent global map.

The need to communicate mapping data between UGVs and the ground control station (GCS) creates many additional problems. Radio signals are often noisy and unreliable, and the challenge required our mapping system to handle periods without communication. Multi-robot mapping systems can either be centralised with a master node at the GCS, or decentralised with each UGV fusing its own version of the global map. In the centralised case a single global map ensures each robot is locked into the same coordinate system, however extended loss of communications to the master node can be fatal. A decentralised design allows UGVs to continue operations individually or in smaller groups, however it is vulnerable to divergence of coordinate systems especially in the absence of any global reference in GPS-denied environments. In this work we describe a hybrid-decentralised solution where mapping is fully decentralised, however a master node may "lock down" parts of the global map to maintain synchronisation.

Diverging or completely unsynchronised UGV coordinate systems are fatal to mission-related tasks such as position commanding and OOI observation correlations. Synchronising coordinate frames and aligning each robots' map is a considerable problem that relates closely to the well known "loop closure" problem described in the SLAM literature [2]. Loop closures occur continuously in multi-robot systems between groups of robots and the global map.

In the decentralised and distributed mapping case, each UGV must process the SLAM data produced by all UGVs in real-time. This requires a highly-efficient approach, especially considering the UGV's processor is already loaded with other intensive mission related tasks. With multiple UGVs operating in the same area, large amounts of overlapping mapping, or occupancy, data is generated. Matching and fusing this data in real-time is a non-trivial problem. Modern graphics processing units (GPUs) have massively parallel stream processing abilities, and inexpensive GPUs are frequently embedded in PCs and mobile devices. In this work we show how they can be leveraged in robotic mapping to accelerate map building and matching tasks. The majority of processing is offloaded onto the GPU, unburdening the CPU.

Since the localisation aspect of SLAM is implicit in multi-robot mapping, we refer to our work simply as a "mapping system". The MAGIC 2010 challenge required a *robust* mapping system for urban environments. It was held at the Adelaide Royal Showgrounds, where the terrain was frequently uneven and included ramps, dirt, sand, and grass. The area did not include stairs or overpasses permitting a 2D mapping

Manuscript received April 20, 2011. This work was supported in part by the Air Force Research Laboratory, under agreement number FA2386-10-4024 U.S. as well as through a grant by Thales Australia.

Robert Reid and Thomas Bräunl are with the Centre for Intelligent Information Processing Systems, School of Electrical, Electronic and Computer Engineering at The University of Western Australia, e-mail: rob@rrfx.net.

approach. Both indoor and outdoor areas required a mapping system that could cope with the transition between full sunlight and indoor lighting and also intermittent civilian-grade GPS availability. The mapping system had to deal with observations of moving OOIs, excluding them and other UGVs from the static global environment maps.

The mapping system described here addresses both these research problems and the challenge requirements. It helped the Western Australian MAGIC robot team (team WAMbot) achieve 4^{th} place in the international MAGIC 2010 challenge. Research contributions include a novel large-scale multi-robot graph-based non-linear map optimisation approach, a hybrid decentralised and distributed mapping system and a novel GPU-based approach to accelerate intensive map matching and fusion operations. The next section reviews previous related work. Section III describes the WAMbot robots and our mapping system in more detail. Section IV presents results from our mapping system in the MAGIC 2010 challenge.

II. Previous Work

The SLAM problem is well understood [1], and many solutions exist for mobile robots fitted with odometry sensors and 2D scanning laser range-finders [2]. We briefly describe the main approaches here, but note that most of these techniques fail to scale appropriately with map size and require some form of hierarchical sub-mapping to deal with a 500m×500m urban environment.

The classical approach is the Extended Kalman Filter (EKF), which represents the map state as a multivariate Gaussian distribution [6]. EKFs scale in $O\left(n^2\right)$ time, where n is the number of features in the map. The dense covariance information typically limits real-time execution to several hundred map features. Sub-mapping approaches exist [7] however heading issues around the origin and the accumulation of non-linearities result in the map eventually becoming inconsistent [6]. While the EKF can be used in multi-robot mapping [5], no convincing results were found that distributed the problem efficiently for large-scale mapping.

The mathematical inverse of the EKF, the sparse extended information filter (SEIF) [8] has been show to scale in linear-time $O\left(n\right)$, however recovering the full map and covariance information for data association is expensive and inconsistencies result from repeated linearisation.

Monte Carlo particle filter (PF) based approaches, such as FastSLAM [9], have demonstrated impressive real-world results. Here the map state is represented by a set of particles that record the historical pose of the robot. Each particle stores one complete hypothetical map that is re-sampled each iteration. Consistency issues [10] and particle depletion ultimately limits the scalability of this approach. Multiple robots have been demonstrated [4], however real-time computational resources bound the total particle count, limiting either the number of robots or the map size and complexity.

Graph-based SLAM approaches store robot pose estimates and observed features, such as laser scans, as nodes in a graph. Constraints between nodes describe the estimated relative transformation between robot poses. Using these constraints a non-linear estimation process jointly optimises the poses of the nodes with the goal to minimise the total error in the constraints. Constraints are typically estimated using a scan-matching approach such as iterative closest point (ICP) [11]. The earliest attempts at graph-based SLAM by Lu and Milios [12] were slow due to inefficient optimisation techniques. Since then many increasingly efficient approaches have been proposed, refer to [13] for a brief review. Graph-based SLAM avoids common inconsistencies by continuously re-linearising the error function.

The recent graph-based SLAM solution by Konolige et al. [13], called "Sparse Pose Adjustment" (SPA), uses a sparse implementation of the Levenberg-Marquardt non-linear optimisation algorithm. It takes into account the full covariance information in the pose constraints and has been shown to execute faster than existing approaches while maintaining lower error bounds. When used incrementally in a typical online optimisation, SPA's computation time increases linearly with the number of constraints. It can incrementally optimise individual new constraints in a 10,000 constraint graph within 10ms on modest hardware [13]. We use SPA in our mapping system, however extend the approach by fusing multiple laser scans into larger "submaps" that become nodes in the SPA graph. Graph-based SLAM approaches are highly-suited to multi-robot mapping [3], since they segment robot pose, sensor data and pose constraints in a way that is easily transferable between individual robots.

There are very few published works describing the use of GPUs in robotic mapping. One notable paper by Olson [14] uses OpenGL GLSL shaders to perform brute-force correlative scan-matching. Results show considerable robustness to initial pose and noise, with exhaustive GPU-based approaches an order of magnitude faster than same calculations on the CPU.

III. Methodology

The WAMbot mapping system runs on each mobile robot (UGV) and the control station (GCS). On each UGV its critical function is to process mapping sensor data, down-sample it and broadcast it in real-time along with localisation information. On each UGV and the GCS it periodically generates complete globally-optimal maps that are output to the path planning and high-level control sub-systems. Each map is a 2D occupancy-grid with 10cm resolution that represents the static environment. They are ortho-rectified, globally-aligned images where each 10cm square pixel records the likelihood that it is free-space, unknown, or occupied [2].

A. Overview

At the heart of the mapping system is a distributed and decentralised graph-based SLAM implementation based on SPA [13]. This non-linear optimisation approach efficiently finds the globally-optimal arrangement of the complete map. Each UGV generates a sequence of small and locally accurate "submaps" containing 2D spatial information describing their surrounds. In this graph-based SLAM formulation submap

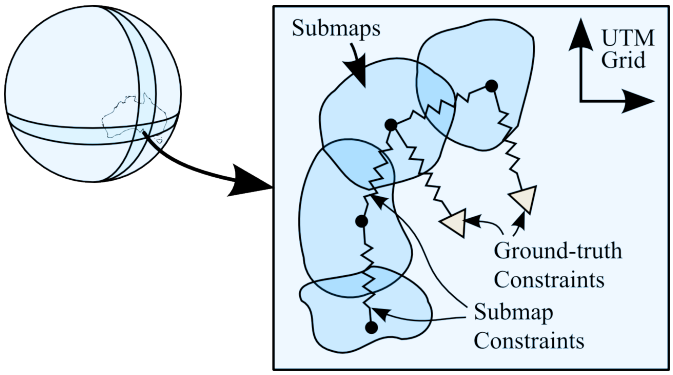

Figure 1. Graph-based multi-robot SLAM using submaps. In a global UTM reference frame submap poses (x, y, ϕ) are optimised to minimise the total error in their spatial constraint "springs".

poses (nodes) are linked by constraints that describe the relative 2D translation and rotation between pairs of submaps. Each constraint includes precision information (inverse covariance) that forms the "springs" describing the strength of the translation and rotation constraints. Figure 1 shows the submap and constraint relationships. Constraints are generated by either relative wheel odometry, global ground-truth measurements, submap matching (loop closure), or added by an operator using the human machine interface (HMI). The mapping system broadcasts compressed submap map data, submap poses, submap constraints and robot poses over-the-air to other UGVs and the GCS. Figure 2 shows the system's logical architecture. It can be separated into SLAM front-end and back-end components:

- *Local SLAM:* the SLAM front-end executes in real-time on each UGV to produce submap information. UGVs broadcast their submap data over DDS.
- *Mapbuilder:* forms the SLAM back-end that runs separately on each UGV and the HMI. It receives all broadcast submap information from UGVs and periodically outputs the global map. The Mapbuilder has 3 sub-components:
  - *Optimiser:* performs incremental map optimisation.
  - *Builder:* composes or fuses the submap data and outputs complete global maps.
  - *Matcher:* searches for spatial matches between submaps creating additional map constraints. It searches for "loop closures" between UGVs.

The following sections detail the UGVs' mapping sensors and each of these software components.

B. Mapping Hardware

Each WAMbot UGV is built on a Pioneer AT-3 robot base and is fitted with a homogeneous sensor suite. The complete WAMbot platform is described in [15]. The primary mapping and localisation sensor for each UGV is a SICK LMS111, a single-plane laser range-finder providing a 20m range and 270° horizontal field of view. The LMS111 is mounted 50cm above the ground where it provides an immediate view of the environment. A second Hokuyo URG-04LX single-plane laser scanner is mounted vertically on the front of each UGV where it measures a profile of terrain elevations. To detect each UGV's local (relative) motion they are fitted with high-resolution wheel encoders and an XSenS MTi inertial measurement unit (IMU) providing 6 degree of freedom (DOF) pose and motion estimates.

Globally-referenced position estimates are generated by a Qstarz civilian-grade GPS receiver, and approximate heading information is provided by magnetometers in the XsenS unit. This relatively "noisy" global pose information is typically only available outdoors with a clear view of the sky. In ideal situations outdoors $\pm$5m position and $\pm 10°$ heading accuracy is achieved. Both the GPS and magnetometers have difficult failure-modes to detect (multi-path reflections and magnetic interference respectively) and are aggressively filtered if the HDOP value rises above 1.5 or satellite count below 8.

Each UGV is fitted with a single Intel Core 2 Duo automotive PC that interfaces directly with these sensors. Wireless networking is based on an ad-hoc consumer-grade 802.11g Wi-Fi overlaid with an OLSR mesh network. Communication is performed at the application level using RTI's fault-tolerant distributed data system (DDS). The mapping system is allocated a maximum 500Kbit/sec total of wireless network bandwidth.

C. Local SLAM

The Local SLAM subsystem provides the SLAM front-end that executes on each UGV. It processes all sensor data in real-time and broadcasts submap data. Refer to figure 2 for logical architecture information. It produces locally metric map tiles, or "submaps" each having their own local coordinate system.

With the sensing characteristics of the LMS111 laser scanner, a single scan can produce a detailed submap that is up-to 40m in diameter and covers a 270° horizontal field of view. The laser beam divergence and noise characteristics (including in sunlight) typically creates a metrically accurate laser point cloud. While the UGV travels across short distances, multiple laser scans taken at 10Hz are aligned using iterative closest

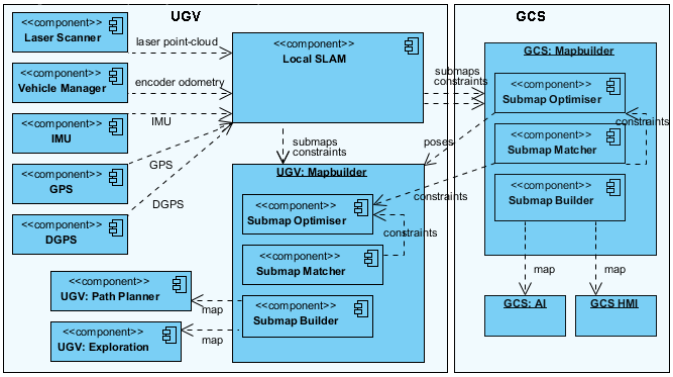

Figure 2. Mapping system Logical Architecture. The components on the left execute as separate processes on each UGV. The Mapbuilder processes executes on each UGV and the on GCS where it outputs complete maps via shared memory where needed. Data is broadcast between UGVs and GCS using DDS over a Wi-Fi mesh network.

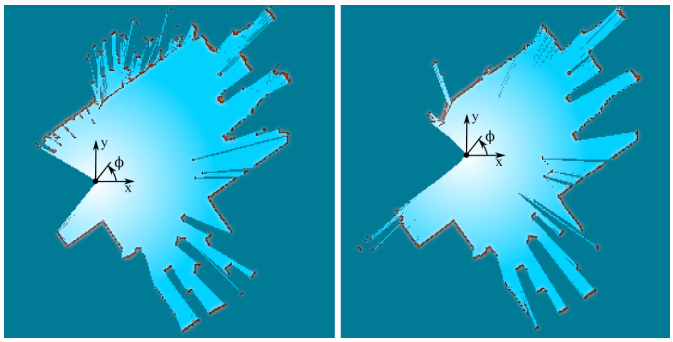

Figure 3. Example 25×25m submaps inside the Old Ram Shed Challenge. Both of the UGVs are at their submap origin facing to the right.

point (ICP) algorithm [11], and aggregated into the current submap. In a typical urban environment the geometry doesn't change significantly over a distance of a few meters, allowing accurate sequential scan registration and very low drift within each submap. Robust outlier rejection (RANSAC) is used to ignore moving objects in the laser data. Figure 3 shows two example submaps from two UGVs.

The ideal mapping system would store and broadcast full laser scan point-clouds and fine-grained odometry between UGVs. However, CPU, storage and wireless bandwidth restrictions require that the sensor data and laser point clouds be filtered and down-sampled. The challenge required accurate navigation of our 50cm wide UGVs through 90cm doorways. As such our approach down-samples the data into 2D occupancy (likelihood) maps with a 10cm grid [2] providing adequate spatial resolution for path planning.

When starting a new submap each UGV's local pose is reset to zero with zero uncertainty. As each UGV navigates through its submap, the relative wheel odometry, IMU data and ICP scan matching results are fused using an extended Kalman filter (EKF). To cope with excessive wheel slippage, UGV tilt information from the IMU is used to switch the EKF between different motion models favouring either odometry or ICP matching.

After moving more than a distance threshold (4 meters in the challenge) the current submap is closed. The UGV's local pose-estimate including its multi-modal Gaussian uncertainty information are broadcast as an odometry constraint joining the recently closed submap with the newly created one. If the GPS and compass sensor is considered valid, this globally-referenced position data is broadcast as a fixed "ground-truth" constraint on the new submap. The submap occupancy-grid data is rendered, compressed and broadcast either every 5 seconds, or when the map is closed, to minimise network traffic. In the challenge environment the transmitted mapping data totalled about 10Kbit/sec per UGV, about 2% of the mapping bandwidth budget per UGV.

D. Mapbuilder

The Mapbuilder provides the SLAM back-end. It periodically optimises the global map and outputs it via shared memory as required. Instances of the Mapbuilder run on each UGV and the HMI. The Mapbuilder has 3 sub-components:

D-1. Submap Optimiser

The submap optimiser implements a multi-robot graph-based SLAM algorithm. It periodically performs an incremental non-linear optimisation on all submap poses for all UGVs using available submap constraints. It is based on the recent sparse pose optimisation (SPA) work by Konolige et al. [13]. We extend their work by decentralising the optimisation process across multiple UGVs and incorporating globally referenced ground-truth constraints (GPS and compass).

SPA's non-linear optimiser uses the Levenberg Marquardt (LM) non-linear optimisation algorithm. For a detailed explanation of the optimisation algorithm and constraint cost function refer to [13]. The submap poses form nodes in a graph and the constraints are the links or "springs" between the nodes. An analogy for the optimisation process is that the submaps are sliding and rotating on a frictionless surface, while the constraint springs of differing strengths pull the submaps into alignment. Ground-truth constraint springs tie the submaps down to globally referenced coordinates (refer to figure 1). The map is globally optimised when the tension in the springs (the constraint error) is evenly distributed.

Keeping multiple completely decentralised maps synchronised is difficult with intermittent communications. While the DDS communications layer is fault-tolerant and includes message buffering and replay, the buffer sizes are limited and in extended operation it is inevitable that some UGVs may lose submap data. To prevent each optimiser from producing different solutions (diverging UGV coordinate systems), an instance of the submap optimiser at the GCS is designated as the "master". We describe the system as a hybrid-decentralised approach. As the master optimiser aligns submap poses, any movements larger than a threshold are broadcast as submap pose updates to the UGVs. The non-master submap optimisers accept these poses and *fix* them in their local map. Partial or complete communication failures with UGVs can occur and they will continue to function independently optimising both new and previously unfixed submaps. When communication is re-established map data is automatically re-synchronised.

D-2. Submap Builder

The submap builder fuses all the submap occupancy-grid information together to create a complete global occupancy-grid map. To efficiently fuse thousands of overlapping submaps, the map is rendered using hardware-based acceleration on the GPU. To ensure cross-platform and hardware compatibility we render the map using OpenGL GLSL fragment shaders into an off-screen render-buffer.

Submaps are held as 8-bit RGB textures in the GPU's memory. The red colour channel stores the occupied/free/unknown cell likelihood information. The green channel holds a Gaussian blurred version of the occupied cells that is used as a cost-map for the navigation system, and the blue channel holds vertical-cost information. Submaps are rendered as OpenGL

"quads" which executes the GLSL fragment shader in parallel using all of GPU's processing elements. Submap RGB data is blended using a cost function and output to the render-buffer after a depth test. The OpenGL depth-buffer is used effectively to fuse submap occupancy data both spatially and temporally. Each pixel's depth value is a cost function based on occupancy value, squared distance from the submap origin, and the time of submap data acquisition. These form an inverted bowl shape in the depth-buffer for each submap, where thousands of them can be rendered without concern of ordering. The depth-buffer blends the final output map which minimises map noise. On modest PC hardware with an entry-level GPU the submap builder can create maps over 800m×800m in size and renders 1000 typical-sized submaps in 120ms with almost no load on the CPU. If the map size exceeds the maximum GPU memory (eg. on an embedded PC), the render is limited to a sliding window and only the necessary submap textures are loaded.

D-3. Submap Matcher

To measure and create constraints between submaps from different robots (multi-robot map fusion), and during loop closure, the Submap Matcher performs an exhaustive search to find the optimum relative pose (x, y, ϕ) that aligns a pair of submaps. On each UGV, the matcher repeatedly attempts to match the current and recent submaps against other submaps within its local area. On the GCS the matcher chooses pairs of submaps based on a heuristic and continuously attempts to create matches (large-scale loop closures).

Spatial matching algorithms typically use an efficient optimisation approach (such as ICP [12]) to reduce computational cost. However they often become trapped in local maxima and fail to provide correct solutions. To provide a robust matching solution we use the "free" processing power available on the GPU to perform an exhaustive correlation search of the 3D solution space (x, y, ϕ). Inspired by Olson's GPU scan-matching approach [14], the submap matching search is made efficient using the hundreds of parallel processing elements available in the GPU.

Referring to the left of figure 4, an OpenGL GLSL fragment shader is executed 9 times over a range of angle increments (eg. $\pm 30°$). In each execution it calculates the correlation between two submap likelihood maps over a range of (x, y) pose offsets (eg. ± 5m). The brute-force search gives a complete survey of the solution-space. Multiple local maxima can be seen in this figure. On a low-end GPU (NVIDIA 9600 GT) about 200,000 submap correlations can be calculated per second. A depth-limited recursive search through existing map constraints identifies Gaussian probabilistic priors that can be combined with the matching likelihood (figure 4, right). The resulting posterior measures both the optimum relative pose and accurate covariance information, which becomes a potential new constraint. The covariance of the pose translation identifies how constrained the matching geometry is. The ratio of the smallest to largest eigenvalue of this covariance matrix is used to scale the matching score: self-similar matches such as corridors are heavily penalised.

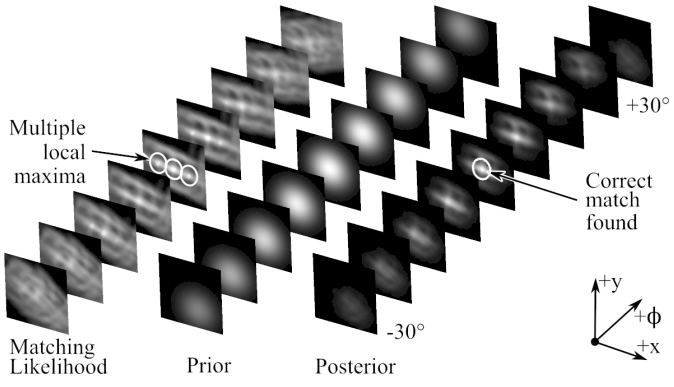

Figure 4. Submap matching. A brute-force GPU-based correlation search rapidly finds local maxima in the relative pose between submap pairs (left). Odometry priors (center) combine to identify the correct relative pose (right).

Potential match constraints are tested in the local optimiser before being accepted. The submap graph is searched to a depth of 2 nodes, and errors in these nearby constraints are evaluated. If the test constraint significantly increases the sum of the constraint errors after a test optimisation it is rejected.

IV. Results

The mapping system has been demonstrated in many environments both indoor, outdoor and also with intermittent GPS availability. It performs as designed, and would easily scale to areas an order of magnitude larger than the challenge area. During the challenge the WAMbot UGVs were troubled by significant hardware problems, and while the mapping system functioned as expected, they collected very little map data. The results presented here were produced by replaying sensor logs into our mapping system in real-time. The sensor logs were provided by the University of Pennsylvania team, who place second in the challenge. Their sensor package was similar to ours and their data was easily reprocessed by our system.

The capabilities of our mapping system are most readily demonstrated by watching the global map evolve over time. We provide two videos[1]. The first from Phase 2 of the challenge, and the second from the Old Ram Shed Challenge (ORSC). The videos play at 15× real-time speed.

The Phase 2 video shows 7 UGVs exploring an agricultural show-ground, passing in and out of horse stables. Figure 5 shows the final map overlaid on aerial imagery. The 200m×160m map is metrically accurate and correctly georeferenced by the intermittent and noisy GPS data. To quantify the RMS error in the final map we selected a set of evenly distributed features in the map and measured their linear error to the aerial image. The estimated mean linear error was 0.57m. The final map has 446 submaps, 419 ground-truth constraints and 1284 submap constraints. For the final map each execution cycle it took 21ms to incrementally optimise the entire pose-graph, and 87ms to build and output the global

[1]Available online at: http://www.rrfx.net/2011/04/large-scale-multi-robot-mapping.html Note: these videos are encoded in DIVX format and play on the freely available VLC media player.

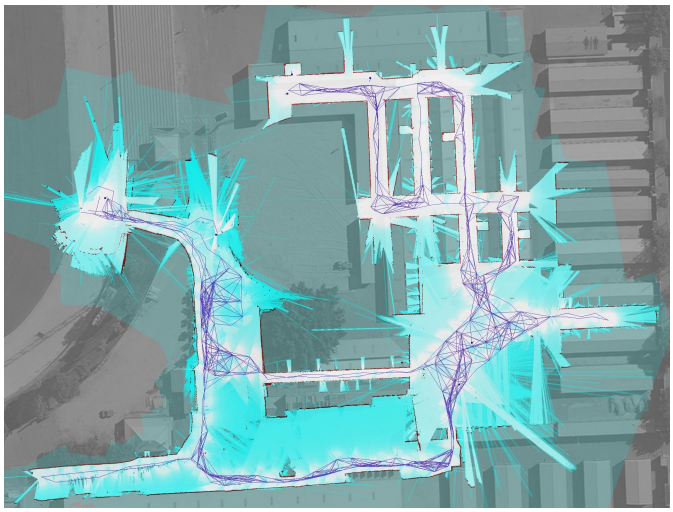

Figure 5. The final global occupancy map for Phase 2 overlaid on aerial imagery. The graph of blue lines represent submap constraints.

occupancy map. On each UGV, with a majority of fixed nodes, the optimisers runs in under 5ms. With a 1Hz execution cycle, the GPU is able to process an additional 7-10 submap matches in the remaining time, more than enough to accurately constrain the map. The initial submaps take some time to become connected due to occlusions and ambiguous geometry in the matching process, however after driving several meters each UGV correctly joins its current submap to the global graph. Slow drifts in submaps are observed as transient GPS noise is filtered into ground-truth data. If additional ground-truth constraints had been supplied by the operator, spatial errors could have been reduced by an order of magnitude.

The ORSC video shows 5 UGVs exploring a 70m$\times$40m agricultural shed. Again the submaps take some time to become connected however once the UGVs begin moving the map is rapidly constrained and is metrically accurate. Note that barriers inside the shed were temporary and very few walls and angles were regular. No ground-truth was made available to evaluate the accuracy of this map. The final map has 260 submaps and 2170 submap constraints. For the final map each execution cycle spent 23ms incrementally optimising the pose-graph and 85ms to build and output the global occupancy map. We note that tunable parameters in the mapping system were not changed when moving between environments.

V. Conclusion

We have described a large-scale multi-robot mapping system that operates in a distributed and decentralised manner. Our system implements an efficient graph-based SLAM approach that readily scales to multiple robots and large environments. It would easily map urban areas an order of magnitude larger than the challenge area. We have shown good performance with relatively inexpensive sensors, integrating both noisy odometry and intermittent civilian-grade GPS data. The system is robust to communications disruptions. Inexpensive yet powerful GPUs are now common and we show how they can be used very effectively for both composing global maps and robust correlation matching of submaps. Our system has been demonstrated in many environments both indoor and outdoor. It performs as designed and helped team WAMbot achieve 4^{th} place in the MAGIC 2010 challenge.

Future work includes porting the WAMbots and our mapping system to ROS running on Linux. The SPA optimisation approach may be replaced by a newer general purpose graph optimisation framework [16]. The GPU-based submap matching and building tasks implemented in OpenGL may be rewritten in the newer OpenCL compute framework. This will allow both mixed-vendor GPUs and newer multi-core CPUs to be easily used by the mapping system.

Acknowledgments

We acknowledge all WAMbot team members for their tireless efforts in the 18 months before and during the challenge. Special thanks to A. Boeing, M. Fazio, A. Gandossi, N. Garel, A. Morgan, F. Ophelders and K. Vinsen. Also thanks to the University of Pennsylvania team, A. Kushleyev, C. Phillips, M. Phillips, J. Butzke and D. Lee for their challenge data.

References

[1] H. Durrant-Whyte and T. Bailey, "Simultaneous localisation and mapping," *Robotics and Automation Magazine*, vol. 13, p. 99–117, 2006.

[2] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. The MIT Press, ISBN: 978-0262201629, 2005.

[3] D. Fox, J. Ko, K. Konolige, B. Limketkai, D. Schulz, and B. Stewart, "Distributed multirobot exploration and mapping," *Proceedings of the IEEE*, vol. 94, no. 7, p. 1325–1339, 2006.

[4] L. Carlone, M. K. Ng, J. Du, B. Bona, and M. Indri, "Simultaneous localization and mapping using Rao-Blackwellized particle filters in multi robot systems," *Journal of Intelligent & Robotic Systems*, 2010.

[5] R. Madhavan, K. Fregene, and L. E. Parker, "Distributed cooperative outdoor multirobot localization and mapping," *Autonomous Robots*, vol. 17, no. 1, pp. 23–39, Jul. 2004.

[6] T. Bailey, J. Nieto, J. Guivant, M. Stevens, and E. Nebot, "Consistency of the EKF-SLAM algorithm," in *International Conference on Intelligent Robots and Systems*, 2006.

[7] P. Piniés and J. D. Tardos, "Scalable SLAM building conditionally independent local maps," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2007, p. 3466–3471.

[8] Y. Liu and S. Thrun, "Results for outdoor-SLAM using sparse extended information filters," in *IEEE International Conference on Robotics and Automation*, 2003, p. 1227–1233.

[9] D. Hahnel, W. Burgard, D. Fox, and S. Thrun, "An efficient FastSLAM algorithm for generating maps of large-scale cyclic environments from raw laser range measurements," in *International Conference on Intelligent Robots and Systems*, 2003.

[10] T. Bailey, J. Nieto, and E. Nebot, "Consistency of the FastSLAM algorithm," in *IEEE Int Conference on Robotics and Automation*, 2006.

[11] F. Lu and E. Milios, "Robot pose estimation in unknown environments by matching 2D range scans," *Journal of Intelligent and Robotic Systems*, vol. 18, no. 3, p. 249–275, 1997.

[12] ——, "Globally consistent range scan alignment for environment mapping," *Autonomous Robots*, vol. 4, no. 4, p. 333–349, 1997.

[13] K. Konolige, G. Grisetti, R. Kummerle, B. Limketkai, and R. Vincent, "Efficient sparse pose adjustment for 2D mapping," in *IEEE/RSJ Int Conference on Intelligent Robots and Systems*, 2010.

[14] E. B. Olson, "Real-time correlative scan matching," in *IEEE International Conference on Robotics and Automation*, 2009.

[15] A. Boeing, T. Braunl, R. Reid, A. Morgan, and K. Vinsen, "Cooperative Multi-Robot navigation and mapping of unknown terrain," in *International Conference on Robotics, Automation and Mechatronics*, 2011.

[16] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "g2o: A general framework for graph optimization," in *IEEE Int. Conf. on Robotics and Automation*, 2011.