2007 IEEE International Conference on
Robotics and Automation
Roma, Italy, 10-14 April 2007

An Efficient Rao-Blackwellized Genetic Algorithmic Filter for SLAM

J.F. Dong, W.S. Wijesoma, and A.P. Shacklock

***Abstract*— A Rao-Blackwellized Particle Filter approach is an effective means to estimate the full SLAM posterior. The approach provides for the use of raw sensor measurements directly in SLAM, thus obviating the need to extract landmarks using complex feature extraction methods and data association. In this paper a solution framework based on Rao-Blackwellized Particle Filters (RB) and Genetic Algorithms (GA) is proposed for recovering the full SLAM posterior using raw exteroceptive sensor measurements, i.e. without landmarks. The resultant Rao-Blackwellized Genetic Algorithmic Filter (RBGAF) permits the uses of any arbitrary measurement model unlike FastSLAM with scan matching. Since the proposed method represents the environmental map state for each robot trajectory using a population of chromosomes as opposed to grids, RBGAF is much more memory efficient than DP-SLAM. Memory efficiency is further enhanced through the exploitation of dynamic data structures for representing the maps and the robot trajectories. This makes the proposed RBGAF very suitable for large scale SLAM in 3D environments. Further, the proposed method's provision for adaptation of chromosome lifetime/group sizes and its ability to incorporate alternative map representations makes it adaptable to varied environments and different sensors. Simulation and experimental results obtained in an outdoor environment using a laser measurement system are presented to demonstrate the method's effectiveness.**

I. Introduction

THE practical objective of simultaneous localization and mapping (SLAM) algorithms is to estimate the joint posterior of robot pose and environmental map consistently and efficiently. Murphy, Doucette and colleagues [1] introduced the *Rao-Blackwellized* particle filter. In this approach particles are used to represent the posterior over some variables, together with other parametric posterior density functions to represent all other variables. This provides for the factorization of the SLAM problem into many separate problems. The framework has since been extended and implemented by many researchers.

Montemerlo [2] proposed a Rao-Blackwellized Particle Filter (RBPF) approach known as FastSLAM to solve for the full SLAM posterior. Here particles are used to represent the posterior over robot paths. The conditional independence of landmarks given the robot pose is exploited to factor the mapping problem into separate problems, one for each landmark in the map. The map features are estimated using Extended Kalman Filters (EKF). This well-known algorithm has since been proven to be a very efficient means of solving the SLAM problem. However, FastSLAM requires reliable feature extraction and data association algorithms. This drawback was later overcome by Hahnel [3]. Hahnel uses the raw laser scans – instead of landmarks – in FastSLAM to avoid feature extraction. Consecutive scans were matched to find the change in robot pose. A parametric model was used to determine experimentally the uncertainty associated with scan matching, and hence the uncertainty in the predicted robot poses. It restricts the implementation of various measurement models and it is imperative that the pose estimation errors from matching scans are independent of the number of observations. It may work provided the experimental environment and the actual environments are very much alike and the number of observations are similar during actual navigation.

Grisetti [4] suggested 'adaptive proposal distribution' and selective re-sampling techniques to reduce the number of particles used in FastSLAM with scan matching. However, the advantages of particle reduction is somewhat outweighed by the method's requirement for high-resolution grids to represent the maps.

Another Rao-Blackwellized approach to SLAM using grid maps is due to Eliazar [5], known as DP-SLAM. Unlike in [4], [5] requires one single grid map. However, in a grid-based approach the memory consumption is quadratic to the length in 2D space and cubic in 3D space. In the extreme case, if one is to exploit the accurate measurements obtained by perfect sensors, the grid resolution must be extremely high and the consequent memory requirements infinitely large, unless artificial noise is added to 'waste' sensor accuracy. In [5], distributed particle mapping is used to greatly reduce the memory consumption and computational cost of copying multiple grid maps. However, in sparse environments a grid map is mostly under utilized due to the sparsity of measurement returns. In sparse 3D space memory inefficiency would be very significant making these methods inapplicable. Furthermore, in real SLAM implementations, the computational cost must be within reasonable limits.

Apart from sample size, sample impoverishment and particle depletion are major issues with particle filter based approaches [6] [7]. Heuristic methods are introduced in [8] to mitigate these problems. Genetic algorithms (GA) have also proved to be effective in this regard. Similarities

J.F. Dong is with the Nanyang Technological University, Singapore. (e-mail: dong0013@ ntu.edu.sg).
W.S. Wijesoma is associate professor with the Division 4, Nanyang Technological University, Singapore. (e-mail: eswwijesoma@ ntu.edu.sg).
A.P. Shacklock is with the SIMTECH, Singapore. (e-mail: andrewps@ SIMTech.a-star.edu.sg).

between GA and particle filters were observed long ago [9]. Techniques in GA have also been applied to mitigate problems in robot navigation with particle filters. For example, crossover operation (modified particle filters) [10], tournament selection [6], and species [7] are applied on particle filters. In [7], chromosomes represent environmental map, but the map is still constructed by features (landmarks).

In this paper, GA techniques and the Rao-Blackwellized particle filters are fused together. Environmental maps are represented by tree-structured groups of child-chromosomes, while robot trajectories can still be represented by ancestry trees as in [5]. Child-chromosomes enable the full use of sensor accuracy without increasing memory consumption; tree-structured groups further reduce memory required to represent the big grid maps by stripping all null space away. Unlike [3], any measurement model can be implemented based on mathematically sound posterior estimation algorithms. The proposed method's built in adaptive chromosome lifetime/group sizes and its ability to incorporate alternative map representations makes it adaptable to varied environments and different sensors.

The rest of the paper is constructed as follows: Section II introduce Rao-Blackwellized Particle Filters. Section III and section IV describe Rao-Blackwellized Particle Filter SLAM and propose the RBGAF-SLAM algorithm. Section V gives a detailed discussion for the computational complexity and memory consumption of DP-SLAM and RBGAF-SLAM. Section VI provides our simulation and experimental results. Section VII gives the conclusions.

II. Rao-Blackwellized particle filters

One of the major drawbacks of particle filters (PF) is that sampling in high-dimensional spaces can be inefficient. In some cases, however, the model has "tractable substructure", which can be analytically marginalized out. Marginalizing out these variables is an example of the techniques called Rao-Blackwellization. Rao-Blackwellized Particle Filters (RBPF) can be implemented in SLAM.

If we were able to sample N i.i.d. random samples (particles), $\{(r_{0:t-1}^{(i)}, g_{0:t-1}^{(i)}), i=1,...,N\}$ at time $t-1$, approximately distributed according to the distribution $p(r_{0:t-1}^{(i)}, g_{0:t-1}^{(i)} \mid y_{1:t-1})$, RBPF allow us to compute N particles $(r_{0:t}^{(i)}, g_{0:t}^{(i)})$ approximately distributed according to the posterior $p(r_{0:t}^{(i)}, g_{0:t}^{(i)} \mid y_{1:t})$, at time t. The detailed descriptions of Rao-Blackwellized particle filters has been introduced in [1].

III. Rao-Blackwellized particle filters SLAM

A. Rao-Blackwellized Particle Filters SLAM

Rao-Blackwellized Particle Filters are an effective means to solve the full posterior of the robot pose. Each particle of $\{(x_{1:t}^{(i)}, m_t^{(i)}), i=1,...,P\}$ consists of one robot trajectory $x_{1:t}^{(i)}$ and one associated environmental map $m_t^{(i)}$. To update the whole particle set $\{(x_{1:t}^{(i)}, m_t^{(i)}), i=1,...,P\}$, both robot trajectory $x_{1:t}^{(i)}$ and map $m_t^{(i)}$ will be resampled (or selected) in pairs. The procedures of simultaneous localization and mapping with RBPF are shown in Fig. 1. The key idea of the approach is to solve the recursive Bayes filter update by the equation:

$$p(x_{1:t}, m \mid z_{1:t}, u_{0:t-1}) =$$
$$p(m \mid x_{1:t}, z_{1:t}, u_{0:t-1}) p(x_{1:t} \mid z_{1:t}, u_{0:t-1}).$$

A particle filter is used to represent robot trajectory $x_{1:t}$ and associate it with a map m, conditioned on each sample of the particle filter. The importance weights of the samples are therefore computed by the likelihoods of the measurements $z_{1:t}$ in the maximum likelihood map constructed by the corresponding robot trajectory. This particular particle has taken. $u_{0:t-1}$ as the control.

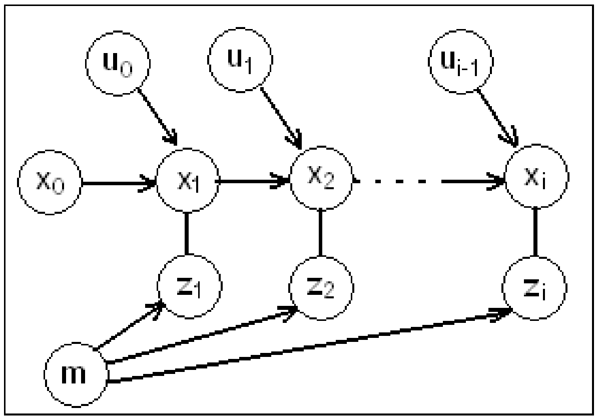

Fig. 1. Graphical model of concurrent mapping and localization

B. Naïve SLAM

In [5], naïve SLAM based on the Rao-Blackwellized Particle Filter is introduced. Each particle consists of one robot trajectory and an associated environmental map in the form of an occupancy grid. Naïve SLAM follows:

```
Take sensor measurements and control inputs $z_{1:t}, u_{1:t}$
for each particle i=1:P;
Predict new robot pose and update robot trajectory based on $p(x_{1:t}^{(i)} \mid x_{1:t-1}^{(i)}, z_{1:t}, u_{1:t})$
Update the occupancy grid map based on observation and new robot pose $p(m_t^{(i)} \mid x_t^{(i)}, z_{1:t}, u_{1:t})$
  for time q=1:t;
    Calculate the particle likelihoods at time q: $L_q^{(i)} = L(x_{1:q}^{(i)}, m_t^{(i)} \mid z_q, u_q)$
  end_for;
```

Calculate weights $w_t^{(i)} = \prod_{q=1}^{t} L_q^{(i)}$

end_for;

Resample new particle sets $(x_{1:t}^{(i)}, m_t^{(i)})$ based on weights $w_t^{(i)}$, denoted as:

$(x_{1:t}^{(i)}, m_t^{(i)}) \sim w_t^{(i)}$

Repeat

In the remainder of this paper, the tilde symbol ~ denotes resampling (or selection) operations.

C. Important issues to be considered

This approach samples approximately at every point in time the full posterior over robot poses and maps. To do this, four key issues must be considered.

- Represent and update P possible robot trajectory (hypothesis).
- Represent maps and update environment map conditioned on robot pose and sensor measurements, respectively.
- Calculate the likelihood, having environment map, trajectory and measurements.
- Resample Rao-Balckwellised particles based on weights calculated.

It has been pointed out that although naïve SLAM is mathematically sound, it is usually not practical. Grid maps consume large memory space, and naïve slam requires copying multiple grid maps during the resampling stage. This could lead to copying gigabytes of memory, which is impractical for real time operations, unless the number of particles (trajectory/map pairs) is very small.

IV. RBGAF-SLAM

To achieve efficient Rao-Blackwellized SLAM and overcome the shortcomings in [3][4][5], this paper presents Rao-Blackwellized Genetic Algorithmic Filter SLAM. The key idea is to bring Genetic algorithm in RBPF so that environment maps are represented by sets of i.i.d. particles (child-chromosomes). In RBGAF-SLAM, each particle of $\{(x_{1:t}^{(i)}, m_t^{(i)}), i = 1, \ldots, P\}$ consists of one robot trajectory $x_{1:t}^{(i)}$ and one associated environmental map $m_t^{(i)}$. The environmental map $m_t^{(i)}$ is represented by a number of child-chromosomes, separated in sub-groups. The jth child-chromosomes in the ith particle (trajectory/map pair) at time t is then denoted as $m_t^{(i,j)}$.

A. Robot trajectory representation and updating

The robot trajectory representations and updating methods used are similar to [5]. New poses are added and the ancestry trees are extended based on the proposal distribution (e.g. motion model) before the resampling process.

B. Environment map representation

In our approach, the environment is represented by tree-structured groups of child chromosomes. Fig. 2 and Fig. 3 show an example of how the environment map is represented by tree-structured groups of child-chromosomes. Chromosomes are separated due to their locations and stored in different sub-groups. Each sub-group is generated when the first observation is made within the particular representation of geometric space.

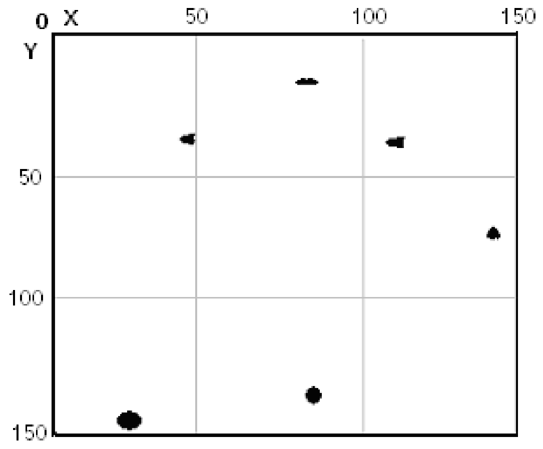

Fig. 2. Environmental map

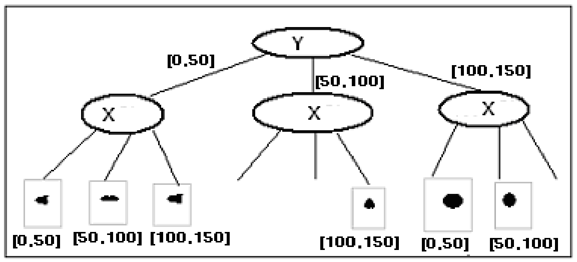

Fig. 3. Tree structured groups

The map representation enables the full use of sensor accuracy without increasing memory consumption as the case in occupancy grids.

C. Environment map chromosomes generation

For each particle i, map chromosomes are generated based on sensor measurements and robot pose.

Generate K_t chromosomes where the number $K_t \propto bn_{zt}$. n_{zt} is the number of observations at time t and b is a constant chosen according to the computational resources. The sub-groups are only generated when observations are obtained within its space. If no chromosomes are to be generated in that sub-group, the sub-group is left empty to save memory space.

D. Environment map updating

The weight of each map chromosome (j=1 to n) is calculated based on its position $m_t^{(i,j)}$ and corresponding robot pose $x_t^{(i)}$ with sensor measurement z_t at time t.

Map chromosomes are then updated by tournament selection within their sub-group. The total number of chromosomes is controlled by chromosome compression.

E. Chromosomes compression

It is very important to control the number of particles. Our solution is to compress extra particles at the map resampling stage. It checks child chromosomes when updating environmental map (tournament selection):

If $dist(m_t^{(i,j1)}, m_t^{(i,j2)}) < \varepsilon$, Compress chromosomes as $(m_t^{(i,j1)}, m_t^{(i,j2)}) \to m_t^{(i,j)} x2$,

in which, $m_t^{(i,j)} = average(m_t^{(i,j1)}, m_t^{(i,j2)})$ and records the count number $c = 2$. Similarly, the child chromosome with count c will be treated as c chromosomes. Record the empty memory slots by a pointer array so that new generated chromosomes will be inserted into these memory slots. Therefore, a maximum number of chromosomes in one group will be assured.

F. Resampling

For particle i, the likelihood is then

$$P^{(i)} = \prod_k P(z_{ik} \mid x^{(i)}, m^{(i)}),$$

where z_{ik} is the measurement between the sensor reading q and particle i, m is the corresponding environmental map.

After the weights are calculated according to the total posterior for each particle $(x_{0:t}^{(i)}, m_t^{(i)})$, the complete chromosome sets are re-sampled. GA techniques are applied to the particles as shown in [6] [7]. The techniques include crossover, mutation, tournament selection and lifespan estimation. Resampling on the particles representing environment maps is not necessary for every time interval and can be processed adaptively to reduce computational cost.

G. RBGAF-SLAM

The algorithm is designed as follows:

```
Take sensor measurements and controls $z_{1:t}, u_{1:t}$
For each particle (trajectory/map pair) i=1:P do:
  Estimate new robot pose and update robot trajectory based on $p(x_{1:t}^{(i)} \mid x_{1:t-1}^{(i)}, z_{1:t}, u_{1:t})$
  Generate j=1:k new map chromosomes $\tilde{m}_t^{(i,j)} \sim p(m_t^{(i)} \mid x_t^{(i)}, z_t)$
  Insert new chromosomes into map $m_t^{(i)} \longleftarrow m_{1:t-1}^{(i)} + \tilde{m}_t^{(i)}$
  Calculate weights for all map chromosomes j=1:k $w_t^{(i,j)} \sim p(m_t^{(i,j)} \mid x_t^{(i)}, z_{1:t}, u_{1:t})$
  Tournament selection, select: $m_t^{(i,j)} \sim w_t^{(i,j)}$
End_for;
If resampling particle (trajectory/map pairs) sets:
For each particle i=1:P do
  Calculate weights: for all map chromosomes j=1:N
  $L_q^{(i,j)} = L(x_q^{(i)} \mid z_q, u_q, m_t^{(i,j)})$
  Calculate likelihood at time q: $L_q^{(i)} = \sum_{j=1}^{N} L_q^{(i,j)}$
end_for;
Calculate weights for i=1:P do: $W_t^{(i)} = \prod_{q=1}^{t} L_q^{(i)}$
Apply tournament selection, select: $(m_t, x_{1:t}) \sim W_t^{(i)}$
End_if;
Repeat
```

V. Computational Complexity

A. Naïve SLAM

If the map is an occupancy grid of size M and P particles are maintained, ignoring the cost of localization, O(MP) operations must be performed merely copying maps. Practically, this approach is naïve in that computational resources are not sufficient to complete this task in most of the cases. Suppose the laser sweeps out an area of intersecting A occupancy grids, it can be found out that A<<M. Therefore, DP-SLAM gives a more efficient way of solving this problem.

B. DP-SLAM

Instead of associating maps with particles, DP-SLAM associates particles with maps. It maintains just a single occupancy grid; each grid square stores a balanced tree. The tree is keyed on the Ids of the particles that have made changes to the occupancy of the square.

Let D be the depth of the ancestry tree for robot trajectories, the computational cost of DP-SLAM can be estimated as:

O (ADPlgP) operations for localization.

O (APlgP) operations to insert new data into the tree.

Ancestry tree maintenance with amortized cost O (ADPlgP)

Because in most of the environments, M>>A and M>APlogP, DP-SLAM will be more efficient compared to Naïve SLAM.

C. RBGAF-SLAM

In RBGAF-SLAM, P particles are maintained. For each particle, a group of child-chromosomes are associated with one robot trajectory. To represent the sensor measurement, n child-chromosomes are required. The number of tree-structured groups is small, having at most 2 to 3 layers. The computational cost of searching particular group is negligible. Then, the following computational complexity can be estimated.

For localization, each particle needs to make O(nDPlgP) accesses to the map. Furthermore, the cost of copying environment maps during the resampling stage can also be estimated as O(nP). The cost of updating the ancestry tree is at most O(DlgP), and can be considered as O(lgP)<<O(nP) as usual case. To resample the environment map, O(nP) tournament selections are needed.

D. *Memory consumption*

In DP-SLAM, when initialising just a single grid map, the memory consumption is at least O(M). To store the entire ancestry tree, the memory consumption is O(ADlgP). If A<<M, O(M) will be the dominating term. In RBGAF-SLAM, to store one single ancestry tree, the memory consumption is O (DlgP); to store all the environment maps, the memory consumption is O(nP). O(DlgP) is negligible compared to O(nP) since n is large.

In an environment where A<<M, especially 3D space, O(M) dominates O(ADlgP) in memory consumption. On the other hand, the actual number of objects observed within 3D environment could be very few. For example, the resolution of current laser range finders measurements in distances is about 5cm; therefore, a 400x400x20m grid map requires $25x10^9$ cells for one single map, which is impractical!! The memory consumption of RBGAF-SLAM, instead, depends on the number of observations but not the dimension of the environment. Therefore, RBGAF-SLAM will be much more efficient in memory consumption in 3D environment.

E. *Summary*

TABLE I. COMPARISON BETWEEN SLAM ALGORITHMS

Algorithm	Raw Sensor Data	Model Sensor noise	Posterior Estimation	Updating time	Memory Consumpt-ion
EKF	NO	YES	Gaussian	Moderate	Moderate
FastSLAM	NO	YES	Gaussian	Moderate	Moderate
Scan-matching FastSLAM	YES	N.A.	Experimental	Moderate	Low
DP-SLAM	YES	YES	ANY	High	High
RBGAF-SLAM	YES	YES	ANY	High	Moderate

In RBGAF-SLAM, our analysis shows that the amortized complexity is O(nPDlgP). O(nP) memory space is required. As discussed, the following table shows the advantages/disadvantages of various SLAM algorithms based on Rao-Blackwellized particle filters and EKF-SLAM.

VI. Simulation and Experimental Results

The simulator represents the environment by *1000x600*, 2-D space, in which the robot starts at the position of (*200,200*), moving rightwards. At the initial stage, 300 particles/trajectories are generated with child-chromosome map associated with each. It is assumed that both bearing and range to objects are made by an exteroceptive sensor. A compass is assumed to provide the orientation information of the vehicle. The noise is assumed to be uniform in the interval -3 to +3 degrees. Wheel encoders have errors with standard derivation 0.5, while the true speed is at a constant of 1 unit/time interval. The environmental map is segmented into *50x50* squares (groups). The mutation rate (*pm*), range (*r*) and the crossover rate (*pc*) are 0.02, 2, and 0.2 respectively.

Fig. 4 shows SLAM simulation in an environment with objects of arbitrary shapes and sizes after the robot has completed a cycle. The area denoted by white is free or empty space and black represents objects that are observed by sensors. The final map is represented by about 1000 chromosomes. Therefore, altogether, about 1000x300 child-chromosomes are generated.

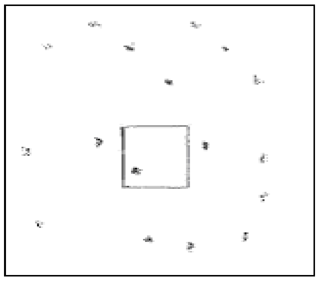
Fig. 4. Simulation results: one complete cycle

Fig. 5 shows the Euclidian positioning error between the estimated robot pose to the real pose and the 3σ bound respecting to the time. Positioning error is small and always lies inside the bound. SLAM consistency is maintained.

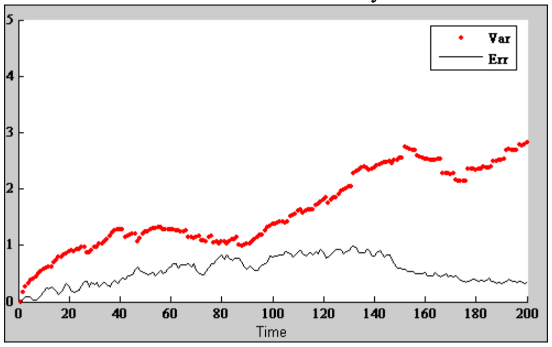

Fig. 5. Errors of SLAM simulation

A Segbot robot (Fig. 6) equipped with laser range finder was tested in the car park of Nanyang Technological University (NTU). By processing the data with RBGAF-SLAM algorithm, both the robot pose and environment map are obtained with satisfactory accuracy. One final map consists of about 800 child-chromosomes; memory consumption is far less than a grid map. Robot trajectories are estimated by 200 particles, associated with one map each.

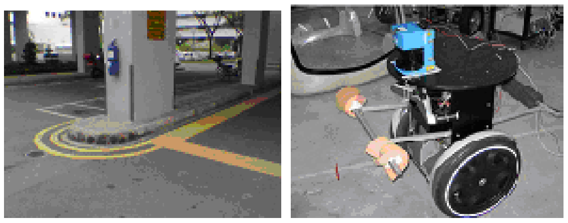
Fig. 6. Experimental environment and Segbot

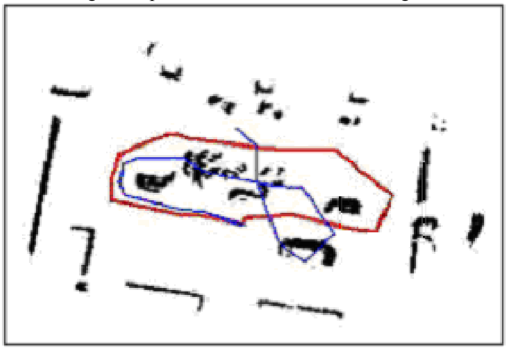
Fig. 7. SLAM in real environment

In Fig. 7, blue line shows the robot trajectory estimated by Odometry, the red line shows the final robot trajectory estimated and black area shows the mapped environment after one complete loop (about 700 updates).

In Fig. 8, the final experimental result of applying RBGAFSLAM on a 1.1km test run with the data set collected from NTU golf cart platform by Mr. Lee, etc. The test run was implemented on the NTU campus, and the loop closing error is quite small as we expected, compared to the campus map.

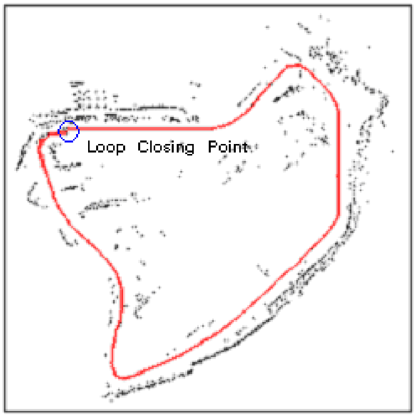

Fig. 8.Complete 1.1km loop in NTU

VII. Conclusions

In this paper, an efficient algorithm for SLAM based on Rao-Blackwellized particle filters and Genetic algorithm filter (RBGAF) was presented for recovering the full SLAM posterior using raw exteroceptive sensor measurements. The use of raw sensor measurements directly obviates the need for complex feature extraction and data association. The algorithmic facilitates the use of any arbitrary measurement model unlike FastSLAM with scan matching. In RBGAF, environmental maps are represented by tree-structured groups of child-chromosomes. Child-chromosomes enable the full use of sensor accuracy without increasing memory consumption, much more memory efficient as compared to DP-SLAM. This advantage makes the proposed methodology very attractive especially in 3D environments. Furthermore, chromosome lifetime and group size can be adaptively adjusted and implemented to control the computational cost below an upper bound; its ability to incorporate alternative map representations makes it adaptable to varied environments and different sensors.

Acknowledgment

We thank Mr. Lee Kwang Wee for providing the 1.1km test data set. We also thank Mr. Tan Chai Soon of the NTU ME-Workshop for his work on the segbot hardware set-up.

References

[1] A. Doucet, J.F.G. de Freitas, K. Murphy, and S. Russel. Rao-blackwellized particle filtering for dynamic Bayesian networks. In Proc. Of the Conf. On Uncertainty in Artificial Intelligence (UAI), 2000.

[2] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. FastSLAM: A factored solution to the simultaneous localization and mapping problem. In AAAI-02, Edmonton, Canada, 2002.

[3] D. Hahnel, W. Burgard, D. Fox, and S. Thrun, An Efficient FastSLAM Algorithm for Generating Maps of Large-Scale Cyclic Environment from Raw Laser Range Measurements, International Conf. on Intelligent Robots and Systems, vol. 1, Page(s): 206 – 211, 27-31, Oct 2003.

[4] G. Grisetti, C. Stachniss, and W. Burgard. Improving grid-based SLAM with Rao-Blackwellised particle filters by adaptive proposals and selective resampling, International Conf. On Robotics and Automation, Barcelona, Spain, 2005.

[5] A. Eliazar and R. Parr. DP-SLAM: Fast, robust simultaneous localization and mapping without predetermined landmarks. In Proc. of the International Joint Conf. on Artificial Intelligence, 2003.

[6] N. M. Kwok, Gu Fang and Weizhen Zhou, Evolutionary Particle Filter: Re-Sampling from the Genetic Algorithm Perspective, IEEE International Conference on Intelligent Robots and Systems, 2005.

[7] Rong Hual L, Binghong H, Coevolutionary Particle filter for Simultaneously localization and mapping, Proceeding of NLP-KE', 2005.

[8] Sebastian Thrun, Dieter Fox, Wolfram Burgard, Probabilistic Robotics, Summer School 2004, pg77-85, 2000.

[9] D. E. Goldberg, "Genetic algorithms in search, optimization and machine learning," Addison-Wesley Pub. Co., Massachusetts, 1989.

[10] N.M. Kwok and G. Dissanayake, Modified particle filter approach for bearing only SLAM, Proceeding on Australasian Conference on Robotics and Automation, 2003.