Preprints of the 2012 20th Mediterranean Conference on Control & Automation (MED), Barcelona, Spain, July 3-6, 2012

G-SLAM: A novel SLAM method

Nikos Zikos and Vassilios Petridis

***Abstract*—Environment perception is a crucial ability for robot's interaction into an environment. One of the first steps in this direction is the combined problem of simultaneous localization and mapping (SLAM). A new method, called G-SLAM, is proposed, where the map is considered as a set of scattered points in the continuous space followed by a probability that states the existence of an obstacle in the subsequent point in space. A probabilistic approach with particle filters for the robot's pose estimation and an adaptive recursive algorithm for the map's probability distribution estimation is presented. Key feature of the G-SLAM method is the adaptive repositioning of the scattered points and their convergence around obstacles. In this paper the goal is to estimate the best robot trajectory along with the probability distribution of the obstacles in space. For experimental purposes a four wheel rear drive car kinematic model is used and results derived from real case scenarios are discussed.**

I. INTRODUCTION

The problem of Simultaneous Localization And Mapping (SLAM) is vital in case of autonomous robots and vehicles navigating in unknown environments [1]. Usually the map consists of a sequence of features (or landmarks), each one of which represents the position of an obstacle or a part of an obstacle (i.e. a big obstacle can be represented by many features).

The Extended Kalman Filter (EKF) was extensively used in the SLAM problem [2], but it has the disadvantage that the computational cost increases significantly with the number of features. Since then, many probabilistic approaches have proposed [3] including the Montemerlo's et al. solution to stochastic SLAM, FastSLAM 1.0 and 2.0 [4], [5], [6], [7], Grid-based SLAM [8], [9], Dual-FastSLAM [10], DP-SLAM [11], L-SLAM [12], etc.

In mobile robots 2-D maps are often sufficient, especially when a robot navigates on a flat surface and the sensors are mounted so that they capture only a slice of the world.

Instead of occupancy grid maps with fine-grained grid defined over the continuous space, in this paper a set of scattered points in the continuous space is used. It is presented a new method, called G-SLAM, where the map is considered as a set of scattered points in the continuous space followed by a probability that states the existence of an obstacle in the subsequent point in space.

A probabilistic approach based on particle filters is used for the robot's pose estimation. In the robot's pose estimation the time series of controls, the time series of the measurements and the latest estimation of the probabilistic map are involved.

A recursive algorithm for the map's probability distribution estimation is used for the map update procedure. The proposed method generates new hypothetical points of features in space which are subsequently tested whether they correspond to real obstacles or not. That is why we call it G-SLAM for Generative-SLAM.

Key feature of the G-SLAM method is the adaptive repositioning of the scattered map's points that results in a convergence of all the points around obstacles. The final map resulted from the G-SLAM method exhibits high density of weighted points around the obstacle and a subsequent high sparsity in the space which is free of obstacles. These weighted points represent the probability distribution of the obstacles in the continuous space. This method fits on problems where a detailed map is needed with low computational resources.

This paper is organized as follows. In section II the model of the robotic system which consist of the robot's kinematic model and the distance-bearing measurement model is described. In section III the probabilistic analysis of the combined SLAM problem in terms of recursively computed probability distributions which estimates the probabilistic map and the robot's trajectory along with the G-SLAM method are presented. In section IV experimental results from real case scenario are discussed.

II. SYSTEM DESCRIPTION

When mobile robots move in an unknown terrain with obstacles, the perception of the robot's environment is crucial in order to interact with it. Thus, a variety of techniques were proposed. Such techniques are the SLAM methods which try to recover the terrain's map and the robot's pose using sensor data from measurements and control inputs. In this paper the map is considered as a set of features $\Theta = [\theta_1, \theta_2, ..., \theta_N]$ each one corresponding to a point in space (fig. 1). The probability that a point θ_k is an obstacle is denoted by p^k. The set of features along with their probabilities is a probabilistic map of the space. The robot's path is represented by a time-series of it's pose $s^t = [s_1, s_2, ...s_t]$. In a planar problem each feature θ_k is a vector with entries (x, y) coordinates of the point. Robot's pose s_t is also a vector with entries (x, y, φ) at time t where φ represents the angle of the robot's orientation corresponding to a global axis system.

A. Planar CarPark

The kinematic equations of a rear drive car-like four-wheel robot are given below.

Nikos Zikos is with the Department of Electrical and Computer Engineering, Aristotle University of Thessaloniki, School of Engineering, Greece (email: nzikos@auth.gr).

Vassilios Petridis is with the Department of Electrical and Computer Engineering, Aristotle University of Thessaloniki, School of Engineering, Greece (email: petridis@eng.auth.gr).

978-1-4673-2529-5/12/$31.00 ©2012 IEEE

$$\begin{bmatrix} p_{x,t+1} \\ p_{y,t+1} \\ \varphi_{t+1} \end{bmatrix} = \begin{bmatrix} p_{x,t} + (v_c\cos(\varphi_t) - (a\sin(\varphi_t) + b\cos(\varphi_t))\frac{v_c}{L}\tan(\omega))\Delta t \\ p_{y,t} + (v_c\sin(\varphi_t) + (a\cos(\varphi_t) - b\sin(\varphi_t))\frac{v_c}{L}\tan(\omega))\Delta t \\ \varphi_t + \frac{v_t\Delta t}{L}\tan(\omega) \end{bmatrix} \quad (1)$$

where v_c is the robot's linear velocity at time t, ω is the steering angle at time t, L is the distance between the car's axles and a, b are the coordinates of the laser sensor according to the car's coordinate system. The velocity v_c is a function of the rear wheel's linear velocity and depends on the steering angle.

$$v_c = \frac{v_e}{1 - \tan(\omega)\frac{H}{L}}$$

where H is the distance between the center point of the rear axle and the rear wheel.

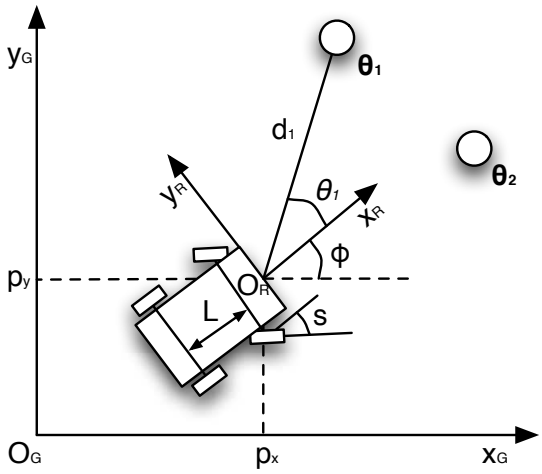

Fig. 1. The robot coordinate system and the position of θ_i landmark with respect to the robot coordinate system O_R. θ is the angle between the vectors X_R and $O_R\theta$

The measurement model of a distance-bearing sensor is given by the nonlinear equations:

$$\begin{aligned} z_t =& g(s_t, \theta_n) = \\ =& \begin{bmatrix} \sqrt{(p_{x,t} - \vartheta_{x,n})^2 + (p_{y,t} - \vartheta_{y,n})^2} \\ \arctan(\frac{p_{y,t} - \vartheta_{y,n}}{p_{x,t} - \vartheta_{x,n}}) - \varphi_t \end{bmatrix} \end{aligned} \quad (2)$$

It is assumed that the distance-bearing sensor's measurements and the control measurements are noisy with noise functions of a known probability distributions.

III. SLAM

SLAM methods are estimating the time series s^t and the map Θ using the observation time series z^t and the control time series u^t. In probabilistic terms this posterior is expressed as:

$$Prob(s^{t+1}, \Theta | u^{t+1}, z^{t+1}) \quad (3)$$

Using the definition of the conditional probability, the posterior in equation 3 can be expressed as:

$$Prob(s^{t+1}|z^{t+1}, u^{t+1}, \Theta) Prob(\Theta|z^{t+1}, u^{t+1}) \quad (4)$$

The two posteriors of the equation 4 correspond to the pose prediction and the map update procedures respectively. The calculation of these two factors is discussed bellow.

A. Pose prediction

The left posterior of the equation 4 refers to the estimation of the pose time series given the map and the time series of the observation and the controls.

$$p_s^{t+1} = Prob(s^{t+1}|z^{t+1}, u^{t+1}, \Theta) = \quad (5)$$

$$Prob(s_{t+1}|s^t, z^{t+1}, u^{t+1}, \Theta) Prob(s^t|z^{t+1}, u^{t+1}, \Theta) \quad (6)$$

In the rightmost posterior the pose s^t is independent of the subsequent measurement z_{t+1} and the control input u_{t+1}. Thus

$$p_s^{t+1} = Prob(s_{t+1}|s^t, z^{t+1}, u^{t+1}, \Theta) Prob(s^t|z^t, u^t, \Theta) \quad (7)$$

which implies the following recursion.

$$p_s^{t+1} = Prob(s_{t+1}|s^t, z^{t+1}, u^{t+1}, \Theta) p_s^t \quad (8)$$

The first factor of the right hand side of the equation 8 using Markov's rule becomes

$$\begin{aligned} &Prob(s_{t+1}|s^t, z^{t+1}, u^{t+1}, \Theta) = Prob(s_{t+1}|s_t, z_{t+1}, u_{t+1}, \Theta) \\ &\propto Prob(s_{t+1}|s_t, u_{t+1}, \Theta) Prob(z_{t+1}|s_{t+1}, s_t, u_{t+1}, \Theta) \end{aligned}$$

The last equation is derived sing Bayes rule. In the first factor, due to the absence of a conditional measure z, the pose s_{t+1} is independent of Θ. In the second factor the conditions s_t and u_{t+1} are redundant due to the Markovian nature of the process (the info of s_t and u_{t+1} are included in s_{t+1}). Therefore the measurement z_{t+1} only dependents on s_{t+1} and Θ.

$$= Prob(s_{t+1}|s_t, u_{t+1}) Prob(z_{t+1}|s_{t+1}, \Theta) \quad (9)$$

In order to calculate the above factorization 9, the simulation technique of the particle filters is used. Thus the target distribution is calculated as:

$$\text{target distr.} = \text{proposal distribution} * \text{importance factor}$$

Through the target distribution the best estimation for the robot's pose s_{t+1} is calculated.

The proposal distribution is generated from the posterior $Prob(s_{t+1}|s_t, u_{t+1})$ as a probabilistic guess of the robot's pose s_{t+1} using the previous pose s_t, the control input u_{t+1} and the robot's kinematic model f.

$$s_{t+1}^i \sim f(s_t^i, u_{t+1}) \quad (10)$$

The importance factor corresponds to the posterior $Prob(z_{t+1}|s_{t+1},\Theta)$ and is calculated directly from the probabilistic map Θ and the subsequent pose s^i_{t+1}

$$w_i = Prob(z_{t+1}|s^i_{t+1},\Theta) \tag{11}$$

Using the measurement model g a guess of the feature $\hat{\theta}^i$ is calculated.

$$\hat{\theta}^i = g^{-1}(s^i_{t+1}, z_{t+1}) \tag{12}$$

From the probabilistic map Θ the feature θ_k that corresponds to the guess $\hat{\theta}^i$ is retrieved. Along with the feature θ_k and its probability p^k which represents the importance factor w_i is retrieved.

In most cases the guess $\hat{\theta}^i$ has not an exact corresponding feature in Θ, thus a technique of bilinear interpolation in the space $\{p,\Theta\}$ is used. Various techniques were tested such as nearest neighbor and bicubic interpolation.

Another case is when observation z_{t+1} corresponds to a new feature. In this case the "nearby space" of $\hat{\theta}^i$ is "almost" empty, thus this observation is not taken into account in the calculation of the importance factor.

Since the set of particles $S^t = \{s^t_1,...,s^t_M\}$ is finite, the "cloud" of particles is growing as the time increases, which can lead to the degeneracy of the algorithm. Thus a resampling technique is necessary. In this paper the technique of Residual Systematic Resampling (RSR) is used [13].

B. Map Update

The rightmost factor of the equation 4 refers to the estimation of the map given the time series of the observation and the controls. The map consists of a set of scattered points in space θ and each one is associated with a probability that the point θ is an obstacle. The distribution of this probabilistic map can be represented by the following posterior.

$$p^k_{t+1} = Prob(z = \theta_k|u^{t+1}, z^{t+1}) \tag{13}$$

The equation 13 gives the probability that the feature θ_k is an obstacle given the observations z^{t+1} and the controls u^{t+1}. By the definition of the conditional probability, the posterior 13 is expressed as:

$$p^k_{t+1} = \frac{Prob(z_{t+1}, z = \theta_k|u^{t+1}, z^t)}{Prob(z_{t+1}|u^{t+1}, z^t)} \tag{14}$$

Using the law of total probability for all θ_j the denominator becomes

$$p^k_{t+1} = \frac{Prob(z_{t+1}, z = \theta_k|u^{t+1}, z^t)}{\sum_j Prob(z_{t+1}, z = \theta_j|u^{t+1}, z^t)} \tag{15}$$

The posteriors of the numerator and the denominator have the same form

$$Prob(z_{t+1}, z = \theta_k|u^{t+1}, z^t) = Prob(z_{t+1}|u^{t+1}, z^t, z = \theta_k)Prob(z = \theta_k|u^{t+1}, z^t)$$

In the rightmost posterior we note that $z = \theta_k$ is independent of the control input u_{t+1} due to the absence of the measurement z_{t+1}. Thus the above expression becomes

$$Prob(z_{t+1}|u^{t+1}, z^t, z = \theta_k)Prob(z = \theta_k|u^t, z^t) = Prob(z_{t+1}|u^{t+1}, z^t, z = \theta_k)p^k_t \tag{16}$$

Equations 15 and 16 imply the recursion:

$$p^k_{t+1} = \frac{Prob(z_{t+1}|u^{t+1}, z^t, z = \theta_k)p^k_t}{\sum_j Prob(z_{t+1}|u^{t+1}, z^t, z = \theta_j)p^j_t} \tag{17}$$

In order to compute the recursion 17 we need to calculate the quantity $q^k = Prob(z_{t+1}|u^{t+1}, z^t, z = \theta_k)$. This quantity can be approximated by:

$$q^k = \frac{1}{2\pi\sqrt{|R|}} e^{-0.5(z_{t+1}-g(\hat{s}_{t+1},\theta_k))^T R^{-1}(z_{t+1}-g(\hat{s}_{t+1},\theta_k))} \tag{18}$$

where $\hat{s}_{t+1}$ is the best estimation of the robot's pose at time $t+1$ and R is the measurements noise covariance matrix.

C. The G-SLAM Method

In this paper the proposed method is based on a technique that generates stochastically new features that are added into the map. Then the update procedure updates the map and removes the "meaningless" features. The stochastic generation of features into the map, is based on the measurement model using the current pose and the current observation. The stochastic nature is due to the sensor's noise. This addition is achieved using a drawing procedure which is described bellow. Afterwards the extended map is updated and the updated features with low probabilities are removed. The small probability in a feature, states that this point in space is unlikely to be an obstacle. Algorithms of this type converges [14]. In the context of this paper, map update procedure converges to high probabilities in features near obstacles. Removing all these features, the parts of space which are free of obstacles are also free of features while on the other hand the parts of space with obstacles gathers all the features.

This method can be described abstractly in four steps:

1) Estimate pose $\hat{s}_{t+1}$ using observation z_{t+1}, control u_{t+1} and the map Θ
2) Generate and add new features θ into the map using drawing process based on the observation z_{t+1}
3) Update map by calculating the probabilities of all features
4) Remove the features with low probabilities

The stochastic addition of new features into the map is achieved based on the observation z_{t+1}. For every observation z_{t+1} a drawing procedure takes place in order to generate a set of M features that represents the sensor's measurement distribution.

$$\hat{z}^i_{t+1} \sim \mathcal{N}(z_{t+1}; z_{t+1}, R_{t+1}) \tag{19}$$

where R_{t+1} is the covariance matrix of the sensor's noise.

Every element $\hat{z}^i_{t+1}$ is given a probability

$$\hat{q}^i = \frac{1}{2\pi\sqrt{|R|}} e^{-0.5(z_{t+1}-\hat{z}^i_{t+1})^T R^{-1}(z_{t+1}-\hat{z}^i_{t+1})} \quad (20)$$

These elements $\hat{z}^i_{t+1}$ are unlikely to correspond to a feature $\theta \in \Theta$. Thus using equation 12 we create new features $\hat{\theta}^i$ and their probability p^i_t in the map Θ is calculated using bilinear interpolation. This procedure augments the probabilistic map with M new features and their approximated probabilities.

After the update procedure, N features with the lowest probabilities p^i_{t+1} are removed. Using this technique it is prevented the overpopulation of the set Θ and the features θ tends to gather near obstacles.

D. Agorithm

A pseudocode of the G-SLAM algorithm is given below.

BEGIN G-SLAM($s_t, z_{t+1}, u_{t+1}, \Theta$)
for $i = 1 : N$ particles **do**
  draw $s^i_{t+1} \sim f(s^i_t, u_{t+1})$
  $w_i = Prob(z_{t+1}|s^i_{t+1}, \Theta)$
  $p^{t+1}_{s,i} = p^t_{s,i} w_i$
end for
Resample p_s
$max(p^{t+1}_s) \rightarrow \hat{s}_{t+1}$
for $i = 1 : M$ new features **do**
  draw $\hat{z}^i_{t+1} \sim \mathcal{N}(z_{t+1}; z_{t+1}, R_{t+1})$
  $\hat{q}^i = \frac{1}{2\pi\sqrt{|R|}} e^{-0.5(z_{t+1}-\hat{z}^i_{t+1})^T R^{-1}(z_{t+1}-\hat{z}^i_{t+1})}$
  $\hat{\theta}^i = g^{-1}(\hat{s}^i_{t+1}, z_{t+1})$
  $\hat{p}^i_t \sim$ Linear interp. of $\hat{\theta}^i$ on space Θ, p_t
end for
for $\theta_k \in \Theta \bigcup \hat{\Theta}$ **do**
  update $p^k_{t+1} = \frac{q^k_t p^k_t}{\sum_j q^j_t p^j_t}$
end for
Remove K features from Θ with the lowest probabilities
END

IV. EXPERIMENTAL RESULTS

For experimental purposes the Car-Park dataset [15] was used. All the experiments were performed on this dataset with the car performing a full loop (fig. 2). A four-wheel rear-drive car was used in this dataset. The car was equipped with a horizontal scanning laser sensor with 80 meters observing radius and 180 degrees field of view. The control vector of this car consists of the linear velocity of the rear wheel and the heading angle of the front wheels. The distance-bearing laser sensor's feedback consists of a 361 distance measurements with half a degree angular distance between them. Also, GPS measurements comes with the dataset, wich were used for the car's position validation.

The algorithm was implemented using the kinematic model of equation 1. The parameters that defines the car's kinematic model are: $L = 2.75$, $H = 0.92$, $a = L + 0.95$ and $b = 0.5$.

The algorithm results in a probabilistic map that consists of a set of points in space and their probabilities of being an obstacle. Figure 2 shows a contour graph with the map's probability distribution. The yellow line represents the best estimation for the car's path.

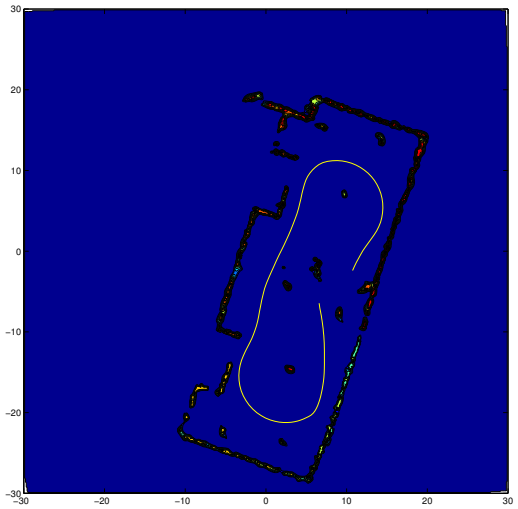

Fig. 2. Contour graph of the map's probability distribution. The yellow line represents the car's path estimation

Figures 2,3,4 demonstrates the map resulted from the G-SLAM method with 8 particles and 10 additional features for every observation, while the resulted map consisted of about 4600 features (a mean density of 2.3 features per square meter).

Figure 3 shows a detailed view of the G-SLAM map probability distribution in comparison to the FastSLAM 2.0 map with 8 particles. It is noteworthy that the G-SLAM's map exhibits more detailed characteristics than the FastSLAM's. Table I shows that the G-SLAM method is slower and inaccurate with small amount of feature particles than FastSLAM, but on the other hand G-SLAM is more accurate and faster when is used with higher number of features M. Also the area which is free of obstacle (blue area) is also free of features θ and all of the features are gathered around the obstacles. The red area represents the highest possibility of the existence of obstacles. Figure 4 shows the surface of the map's probability distribution on the same run.

Experiments performed with a variety in the number of particles N and in the number of added features M. Table II presents the resulted mean distance error of the car and the mean process time using 2, 8 and 30 particles in the pose estimation procedure and 4, 10 and 20 additionally generated features for every observation in the map update procedure. In all experiments the removed features were 30% of the added feature M.

Table II shows that the algorithm results in a relatively high mean distance error when runs with 2 particles, due to its incapability to be consistent with few particles. In this case the map and the car's path acquire a cumulative error high

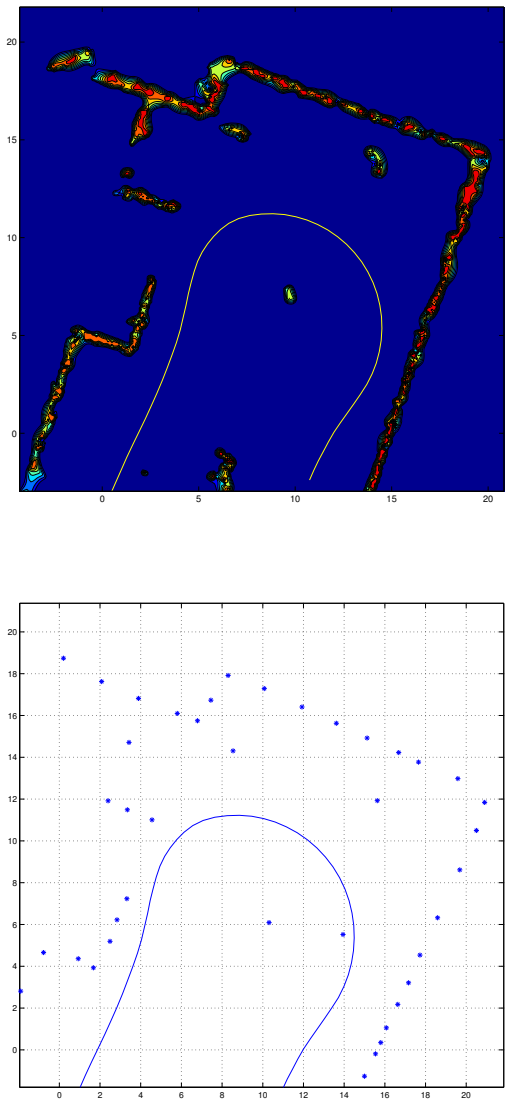

Fig. 3. A detailed contour graph of the map's probability distribution in contrast with the FastSLAM 2.0 map with 8 particles. Colors blue to red corresponds to low to high probability. The yellow line represents the car's path estimation

enough to lead the algorithm into inconsistency. On the other hand the algorithm seems to converge relative fast with respect to the number of particles, since with 8 particle results in the minimum error. It is noteworthy that the process time is independent of the number of particles used. This results from the fact that the pose prediction procedure is much less computational costly than the map update procedure. On the other hand time is almost proportional to the number of the additional features per observation.

V. CONCLUSIONS AND FUTURE WORKS

In this paper it is presented the G-SLAM method for the estimation of the SLAM problem. This method is based on the simulation technique on both the kinematic and measurement models. Combining probabilities resulted from recursive forms the algorithm exports a detailed probability distribution of the map along with the best estimation of the robot's trajectory.

Future work will be the extension of the G-SLAM method in to dynamic environments. The method and techniques we have developed will be applied to a robotic platform and we will investigate the accuracy of the results and the consistency of the method in real case scenarios and dynamic environments.

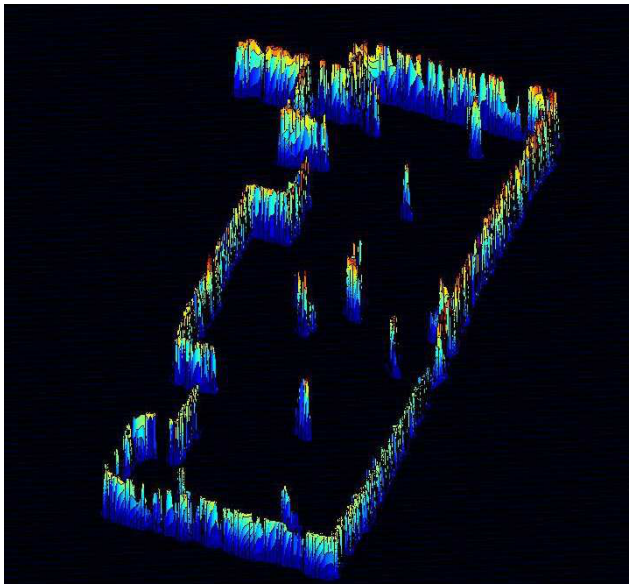

Fig. 4. Surface of the map's probability distribution as resulted from the G-SLAM method

TABLE I
COMPARISON RESULTS BETWEEN G-SLAM, FASTSLAM 1.0 (FS 1) AND FASTSLAM2.0 (FS 2) ON THE CAR PARK DATASET

Method	Number of particles N	Number of features M	Time/step (sec)	Position error (m)
GSLAM	2	4	42 ms	1.55 m
FS 1	2	–	12 ms	0.84 m
FS 2	2	–	31 ms	0.50 m
GSLAM	8	10	107 ms	0.41 m
FS 1	8	–	51 ms	0.62 m
FS 2	8	–	101 ms	0.42 m
GSLAM	30	10	107 ms	0.40 m
FS 1	30	–	148 ms	0.43 m
FS 2	30	–	281 ms	0.40 m

REFERENCES

[1] R. Smith, M. Self, and P. Cheeseman, "A stochastic map for uncertain spatial relationships," in *Proceedings of the 4th international symposium on Robotics Research*. Cambridge, MA, USA: MIT Press, 1988, pp. 467–474.

[2] R. Smith and P. Cheeseman, "On the representation and estimation of spatial uncertainty," 1986.

[3] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents series)*. The MIT Press, 2005.

[4] M. Montemerlo and S. Thrun, "Simultaneous localization and mapping with unknown data association using fastslam," in *Robotics and Automation, 2003. Proceedings. ICRA '03. IEEE International Conference on*, vol. 2, Sept. 2003, pp. 1985–1991 vol.2.

TABLE II
COMPARISON RESULTS ON THE CAR PARK DATASET WITH DIFFERENT NUMBER OF PARTICLES AND DIFFERENT NUMBER OF ADDED FEATURES

Number of particles N	Number of features M	Time/step (ms)	Position error (m)
2	4	42 ms	1.55 m
2	10	107 ms	1.15 m
2	20	228 ms	1.19 m
8	4	42 ms	0.44 m
8	10	107 ms	0.41 m
8	20	228 ms	0.40 m
30	4	42 ms	0.42 m
30	10	107 ms	0.40 m
30	20	228 ms	0.40 m

[5] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "Fastslam: A factored solution to the simultaneous localization and mapping problem," in *AAAI/IAAI*, 2002, pp. 593–598.

[6] M. Montemerlo, S. Thrun, D. Koller and B. Wegbreit, "Fastslam 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges," in *IJCAI*, 2003, pp. 1151–1156.

[7] M. Montemerlo, S. Thrun, and B. Siciliano, *FastSLAM: a scalable method for the simultaneous localization and mapping problem in robotics*. Berlin: Springer, 2007, vol. v. 27.

[8] M. Dissanayake, P. Newman, S. Clark, H. Durrant-Whyte, and M. Csorba, "A solution to the simultaneous localization and map building (slam) problem," *Ieee Transactions On Robotics and Automation*, vol. 17, no. 3, pp. 229–241, June 2001.

[9] G. Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with rao-blackwellized particle filters," *IEEE Transactions on Robotics*, vol. 23, no. 1, pp. 34–46, 2007.

[10] D. Rodriguez-Losada, P. San Segundo, F. Matia, and L. Pedraza, "Dual fastslam: Dual factorization of the particle filter based solution of the simultaneous localization and mapping problem," *Journal of Intelligent and Robotic Systems*.

[11] R. P. Austin Eliazar, "Dp-slam: Fast, robust simultaneous localization and mapping without predetermined landmarks," in *International Joint Conference on Artificial Intelligence*, 2003.

[12] V. Petridis and N. Zikos, "L-slam: reduced dimensionality fastslam algorithms," in *WCCI*, 2010, pp. 2510–2516.

[13] N. Kwak, G.-w. Kim, and B.-h. Lee, "A new compensation technique based on analysis of resampling process in fastslam," *Robotica*, vol. 26, no. 2, pp. 205–217, 2008.

[14] A. Kehagias, "Convergence properties of the lainiotis partition algorithm," *Control and Computers*, vol. 1, p. 6, 1991.

[15] J. Nieto, J. Guivant, E. Nebot, and S. Thrun, "Real time data association for fastslam," in *Robotics and Automation, 2003. Proceedings. ICRA '03. IEEE International Conference on*, vol. 1, Sept. 2003, pp. 412–418 vol.1.