Sādhanā Vol. 40, Part 5, August 2015, pp. 1385–1433. © Indian Academy of Sciences

Where am I? Creating spatial awareness in unmanned ground robots using SLAM: A survey

NITIN KUMAR DHIMAN[1,*], DIPTI DEODHARE[1] and DEEPAK KHEMANI[2]

[1]Centre for AI & Robotics, Bangalore 560093, India
[2]Department of Computer Science and Engineering, Indian Institute of Technology Madras, Chennai 600036, India
e-mail: nitinkdhiman@gmail.com; dipti@cair.drdo.in; deepak@iitm.ac.in

MS received 16 January 2014; revised 11 September 2014; accepted 10 April 2015

Abstract. This paper presents a survey of Simultaneous Localization And Mapping (SLAM) algorithms for unmanned ground robots. SLAM is the process of creating a map of the environment, sometimes unknown *a priori*, while at the same time localizing the robot in the same map. The map could be one of different types *i.e. metrical*, *topological*, *hybrid* or *semantic*. In this paper, the classification of algorithms is done in three classes: (i) Metric map generating approaches, (ii) Qualitative map generating approaches, and (iii) Hybrid map generating approaches. SLAM algorithms for both static and dynamic environments have been surveyed. The algorithms in each class are further divided based on the techniques used. The survey in this paper presents the current state-of-the-art methods, including important landmark works reported in the literature.

Keywords. SLAM, graph-based SLAM; metrical SLAM; topological SLAM; hybrid SLAM; dynamic SLAM.

1. Introduction

An autonomous mobile robot has the capability to perform a class of pre-designated tasks on its own. Autonomy is desirable across various functionality levels. It is necessary for processing sensor data to get information of the environment, to formulate its plan to achieve the given goals, to plan a path for movement to a designated location, to allow the robot to update itself continuously to incorporate observed changes in its plans, *etc.* Among these activities, autonomous movement is very important for mobile autonomous behaviours. A robot has to reach the desired location before performing any activity there. Planning to move to a specific location involves the following three activities: (i) path planning, (ii) localization and (iii) mapping.

Localization is the problem of estimating the pose of the robot relative to a map. Proper localization allows a robot to verify its current status *w.r.t.* the movement plan. This is needed

*For correspondence

because relying purely on odometry is fraught with errors. For example, in case of wheeled robots, odometry fails to capture slippages of wheels. A Global Positioning System (GPS) sensor can also be used for localization, but it could have errors in order of meters. The use of a Bayesian filter *e.g.* Kalman Filter (Kalman 1960) and its variants, and Particle Filters (Doucet *et al* 2001) for combining the output of the GPS sensor along with the inertial navigational system (INS) and odometry, provides a good hybrid mechanism for estimating the location of the robot with minimal errors.

Mapping is the process of creating a spatial model of the environment. Once the robot has acquired a map of the environment, it updates new observations in the map *w.r.t.* its current location. But since the robot also uses this map to obtain its location, and localization is error-prone, the resulting map could also have errors. Thus simultaneous localization and mapping (SLAM) (Bailey *et al* 2000) becomes a chicken-and-egg problem. In the literature, the concurrent execution of these two tasks has also been known as concurrent mapping and localization (CML). However, SLAM is now a more popular and a standard term.

Figure 1 depicts the interaction between these three tasks a robot needs to solve for autonomous movement. Autonomous navigation demands simultaneous planning, localization and mapping (SPLAM). For path planning, a robot should know what its location is and should have access to a map of the environment. Hence, SLAM is fundamental for autonomous navigation (Dhiman *et al* 2012) in an unknown environment.

To perceive the environment, a robot should have adequate sensors and the capability to process sensor data. Like any other physical system, these sensors are also prone to noise which leads to uncertainties at various decision points. Errors in measurements due to noise are not statistically independent; errors keep adding up as the number of measurements increase. This leads to the *correspondence problem* or the *data association problem*. The problem lies in determining whether measurements taken at different times are of the same physical place or object in the environment. The correspondence problem can also arise because of perceptual similarities because of repetitive patterns in the environment, partial visibility of the scene because of occlusions or limited range of the sensor, the change in the dynamic environment over time, *etc*. It can also be a result of the uncertainties introduced while processing sensor data. The correspondence problem can lead to *loop-closure problems*, wherein the robot is unable to recognize that it is passing through an earlier visited place, and thus moving on a loop.

This paper presents a survey of algorithms and technologies for localization and mapping. This paper is an extension of our earlier work (Dhiman *et al* 2012). Most of the state-of-the-art

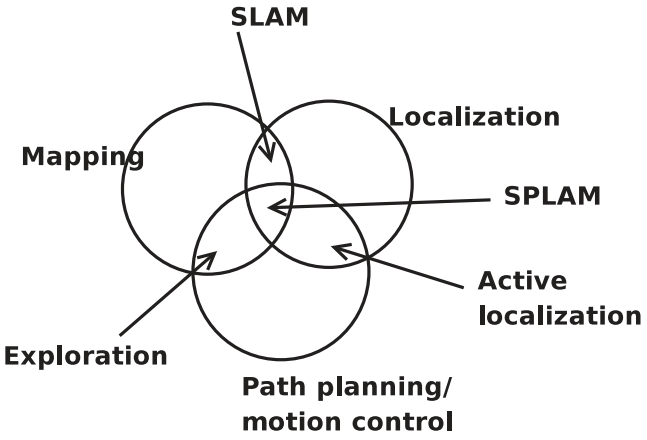

Figure 1. Tasks to be solved by robot for autonomous behaviour (Stachniss 2009; Makarenko *et al* 2002).

SLAM approaches assume a static environment. In a static environment, nothing changes in the world when the robot is operating. SLAM is considered as a solved problem for static environments. For any real world operations, environment is always dynamic. For SLAM in any dynamic environment, the challenge is to keep track of moving and potentially movable entities, and to decide whether to consider such entities for SLAM or not. In the following section, a classification of the various approaches for solving the SLAM problem is presented. The section also provides the roadmap to the rest of the paper in the context of this classification.

2. Classification of SLAM techniques

The output of a SLAM algorithm is a map and the location of the robot in the same map. The map could be *metrical*, *topological*, *hybrid* or *semantic*.

A metrical map can represent the environment as a grid map or a feature map or a geometric map (Stachniss 2009). A landmark map represents the environment by specifying the exact metric position of the relevant landmarks, features, *etc.* The features are encoded *w.r.t.* a single metric frame of reference. The choice of features to use is influenced by the choice of sensors used. Typically, features such as corners or lines are used for range sensors. More complex features such as FLIRT (Fast Laser Interest Region Transform) (Tipaldi & Arras 2010) are also being used. In the case of vision sensors, SIFT (Lowe 2004), SURF (Bay *et al* 2008), Bag of Words methodologies (Cummins & Newman 2008), *etc*. have been used. In other applications it could be observed radio signal strength (Kleiner *et al* 2007; Huang *et al* 2011). In some application environments, such as mines and disaster areas (Kleiner *et al* 2007), the contamination of the environment with artificial landmarks such as RFID may be allowed. The use of such artificial landmarks can ease the burden of solving the correspondence problem in the mapping process (Forster *et al* 2013). An example of landmark based map is shown as figure 2a. A grid represents the environment as a grid of equal size cells. Each grid-cell is a representative sample of the environment. The most used value for the grid cell is the probability of the cell being occupied by an obstacle (Stachniss 2009). Such volumetric grid maps are also called *occupancy grid maps*. An example of occupancy grid map is shown as figure 2b. A geometric metrical map represents the environmental obstacles as geometric shapes *e.g.* circle and ellipse. The geometric

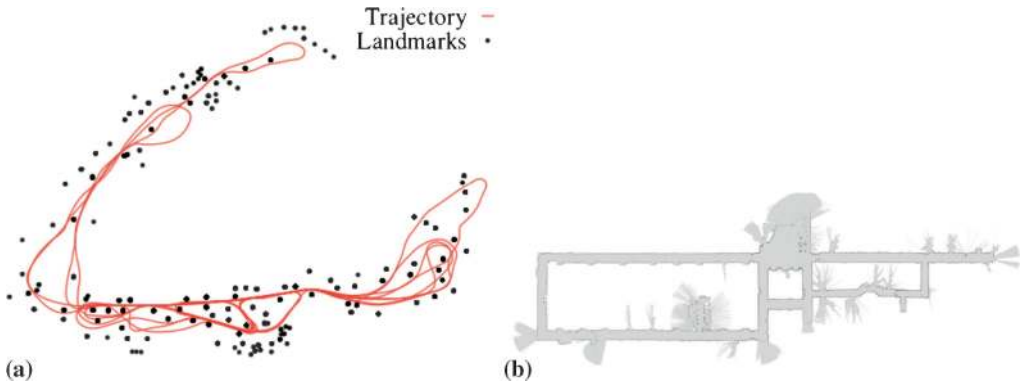

Figure 2. Example of metrical maps. (**a**) The output of g^2o (generalized graph optimization) (Kümmerle *et al* 2011) algorithm on Victoria Park data set. (**b**) The output of DP-SLAM (Eliazar & and Parr 2003) on Wean Hall data set. In this figure, black, gray and white colours represent obstacles, traversable area and unexplored area respectively.

map only models the obstacles in the environment, thus it needs less memory for storage. On the other hand, volumetric maps require a lot of memory storage. Also, the accuracy of volumetric maps depends on the discretization resolution as instantiated by the size of the grid-cell. The advantage of using grid based maps is that a constant time access to the environment is possible.

A *topological* map describes the environment as a graph. The nodes in the graph represent locations, objects of interest, *etc*. The edges express the relationship between nodes which maybe topological (left, right, front, *etc.*), mereological (part-of relationships) or just representing connectivity. An example of this kind of map is the Metro-rail line link maps or subway-maps. Such maps tell a user at which station one metro rail line meets another metro rail line, to which other stations is a particular station is directly connected to, *etc*. A second example is a topological map of rooms in a building where each node is a room and edge defines the adjacency. Topological maps do not provide metrical information. These are usually more compact than metrical maps, and thus require much lesser memory for storage. An example of a topological map is shown in figure 3a. In this example, the nodes (red circles) in the map represent the distinct places in the map and the edges (dashed red lines) in the map represent the connectivity among the nodes. Topological maps are useful for high level planning activities where the metrical information is not required. For example, output of such a planning task, evolved based on a topological map could be 'stay to the left of point A and move along the edge connecting point B and point C'.

A *hybrid* map consists of both topological and metrical knowledge. For example, some metrical maps are connected via a topological relationship. Each node in hybrid map can be either a small metric map or contain some qualitative information about some place or both. The advantage of hybrid maps is that depending on the usage, information at different granularity levels can be accessed. An example of a hybrid map is shown in figure 3b.

Semantic maps provide semantic information associated with the elements of the map *e.g.* labelling of objects like cup, saucer, bottle, table, building, lamp-post and tree, information about relationship between objects, properties of objects like colour, *etc*. The advantage of semantic maps is that they allow interaction with the robot at a conceptual level. For example, with the availability of a semantic map, a command to the robot could be, 'take the second left turn after the green building'.

Various other classifications (Thrun & Leonard 2008) of SLAM are: (i) full SLAM vs online SLAM, (ii) any-time SLAM vs any-space SLAM, (iii) active SLAM vs passive SLAM, (iv) offline SLAM vs online SLAM. A full SLAM estimates the map and the entire path traversed by the robot on this estimated map. Online SLAM algorithms estimate the most recent pose of the robot in the context of the estimated map. While both online SLAM and full SLAM algorithms estimate the map, online SLAM algorithms estimate only the current pose of the robot *w.r.t.* the map estimated, on the other hand full SLAM algorithms estimate the full trajectory traversed by the robot based on the history of the robot poses. In Active SLAM, the robot drives itself autonomously, in an exploratory manner, to acquire data of the environment, while in Passive SLAM the robot is manually driven around. *Any-time* and *any-space* algorithms can operate in resource-constraint robots, and depending on resource availability can demostrate graceful degradation in the output. Any-space SLAM algorithms will work irrespective of the available memory size. Any-time SLAM algorithms will return the best possible map that can be generated based on the sensor observations in a specified amount of time. A mapping approach may be *offline* or *online*. In online mapping approaches, the sensor data is included for map generation as soon as it is observed. For fast responses in real environments, it is desired that the existing map is augmented with the latest observed data rather than creating the map again from scratch. The updated map is used for all future purposes by the robot. In the offline mapping approaches, as a

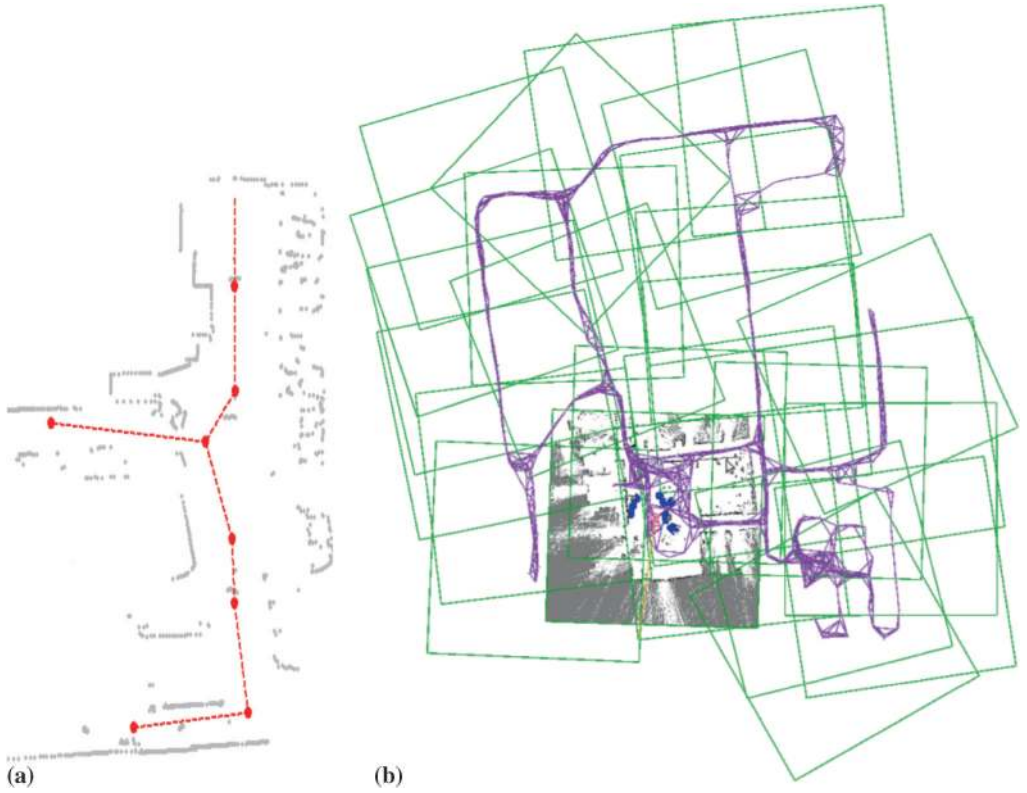

Figure 3. (**a**) An example of topological map (Tapus 2005). Red circles are nodes in the map. Edges are represented as dashed red lines, depicting the connectivity among nodes. A topological map is overlaid over a metric map for ease of visualization. (**b**) an hybrid map (Marder-Eppstein *et al* 2011). Each node in topological map is an occupancy grid map. Each node is shown as green square. The square grid with obstacle information is the current active grid. The graphical map is shown in purple colour. An open-source implementation for this algorithm is available as a package named topological_navigation with Robot Operating System (ROS) (ROS 2014).

first step, all the data is collected. The robot is tele-operated or moved through some pre-planned path. Consequent to data acquisition, the complete dataset is processed in the batch mode.

SLAM using vision sensor is known as Visual SLAM (VSLAM). VSLAM is a problem identical to Structure from Motion (SfM) and Bundle Adjustment (BA). An environment can be represented as a collection of observations from a vision sensor *i.e.* a set of images. An image can be represented as a collection of visual features. There is a high probability that many images in this set have common environment observations, and hence common features. Therefore, the collection of these visual features in the set of images provides an abstract representation of the environment. VSLAM aims to estimate the correct 3D location of these features, along the location of the camera from where the images are captured. SfM is essentially the same task. SfM is an offline activity and a batch-operation is executed on the given collection of images. Real-time SfM is VSLAM. VSLAM may have access to more information than SfM as the robot's wheel odometry may be available. This will provide some estimate of the camera position to the VSLAM algorithm. Bundle Adjustment is a batch optimization procedure on the stream of

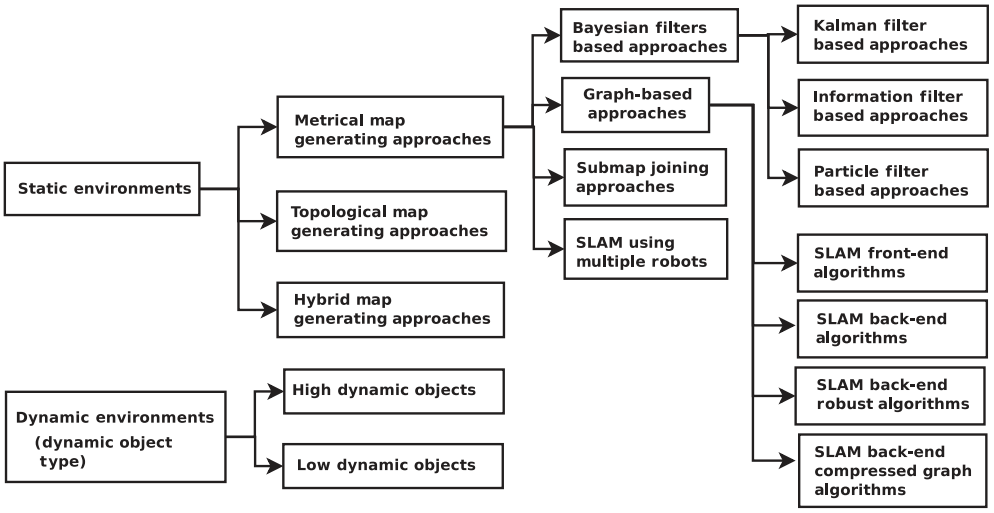

Figure 4. The classification hierarchy of the SLAM approaches presented in this paper.

selected images. The aim of BA is to find 3D coordinates of each point feature along with relative motion between images and the characteristics of the camera from which images are captured. VSLAM, SfM and BA refer to the same problem. A recent survey on SLAM techniques using vision sensors is presented in (Fuentes-Pacheoco *et al* 2012). Similary, a recent survey on SLAM techniques using vision sensors for driverless cars is presented in (Ros *et al* 2012). Hence, this paper does not cover SLAM methods specific to vision sensors in detail.

A number of surveys and tutorials have been presented. Thrun (2002b) has presented a survey on initial SLAM techniques. A tutorial on SLAM and its aspects is presented by Durrant-Whyte & Bailey (2006) & Bailey & Durrant-Whyte (2006). This paper focuses on the general methods and state-of-the-art techniques for SLAM. This paper also does not cover semantic maps. Figure 4 summarizes the classification presented in this paper. The following sections describe various map generating approaches. SLAM algorithms for metric, topological and hybrid maps are discussed in Sections 3, 4 and 5 respectively. In Section 6, SLAM algorithms for dynamic environments are presented based on the type of dynamic objects. High dynamic objects are entities which move frequently and should not be part of the map *e.g.* cars and moving people. These objects should be identified and removed during the mapping process. Low dynamic objects are entities which should be part of the map and can help the robot in localization. Such objects move with low frequency or are so slow that their movement is invisible to the robot *e.g.* door (open or closed), furniture, and items stored in a warehouse.

3. Metrical map generating approaches

Most of the state-of-the-art solutions for metrical maps are probabilistic in nature. They use probabilistic models of the environment, the robot and its sensors. They rely on probabilistic inferencing mechanisms for transforming the sensor outputs into the map. Probabilistic techniques play an important role because they provide established methods for modelling sources of noise and its effect on the model of the environment, and the robot pose. These approaches mostly focus on feature-based map building and more effectively work for a *small-scale space*.

As explained in Kuipers *et al* (2000), this paper distinguish between a *large-scale space*, with spatial structure larger than the agent-sensory horizon, and a small-scale space, with structure within the sensory horizon. The following subsection will discuss approaches based on Bayesian Filters *e.g.* Kalman Filter (Kalman 1960) and its variants, Particle Filters (Doucet *et al* 2001), Information Filters (Thrun *et al* 2006) and graph-based approaches.

3.1 *Bayesian filter based approaches*

The aim of the Bayesian filter based approach is to formulate the SLAM problem as a system state estimation problem. Here, the system's state refers to the robot's pose(s) and location of landmarks detected till the current time. Figure 5 represents a mobile robot moving through the environment. We define the following quantities at time t to facilitate the discussion (Durrant-Whyte & Bailey 2006):

- x_t: the robot pose (the location and the orientation). x_0 represent the starting position of the robot.
- u_t: the control vector applied at x_t to reach x_{t+1}.
- m_k: the location of k^{th} landmark.
- z_k: a vector representing observations made by the robot at time t. An individual observation *i.e.* an observation for landmark i, will be denoted as z_{it}.
- $X_{0:t} = \{x_0,\ x_1, ...,\ x_t\} = \{X_{0:K-1}, x_k\}$ represents the trajectory of the robot till time t.
- $U_{0:t} = \{u_1, u_2, ..., u_t\}$ is set of all control inputs.
- $m = \{m_1, m_2, ..., m_n\}$ is the map, the set of all landmarks' positions.
- $Z_{0:t} = \{z_1, z_2, ..., z_t\}$ is the set of all observations.

In probabilistic formulation of SLAM, the following probability distribution is required to be computed at time t

- For online SLAM: $P(x_t, m|Z_{0:t}, U_{0:t}, x_0)$
- For full SLAM: $P(X_{0:t}, m|Z_{0:t}, U_{0:t}, x_0)$

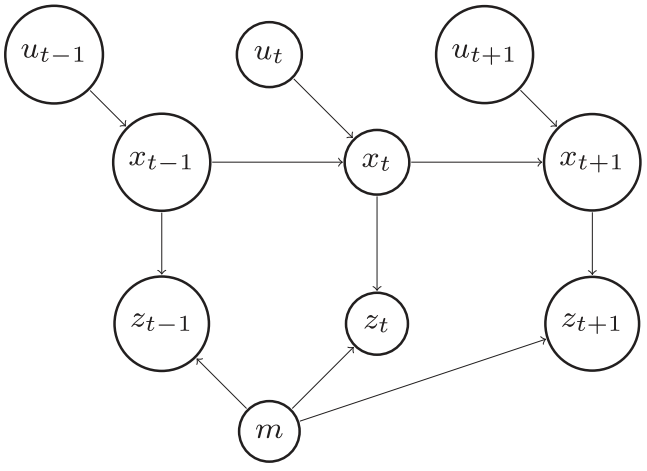

Figure 5. Graphical model for Online SLAM. The robot moves from x_{t-1} to x_t using motion command u_{t-1}. At x_{t-1} position, the robot observes the z_{t-1} landmarks.

Using Bayesian filters, estimation of these probabilities can be implemented as a two step recursive process, namely (i) Prediction step, (ii) Correction step. For online SLAM, the Prediction step is

$$P(x_t, m|Z_{0,t-1}, U_{0:t}, x_0) = \int P(x_t|x_{t-1}, u_t) * P(x_{t-1}, m|Z_{0:t-1}, U_{0:t-1}, x_0)\mathrm{d}x_{t-1} \quad (1)$$

and the Correction step is

$$P(x_t, m|Z_{0:t}, U_{0:t}, x_0) = \frac{P(z_t|x_t, m)P(x_{t-1}, m|Z_{0:t-1}, U_{0:t-1}, x_0)}{P(z_t|Z_{0:t-1}, U_{0:t-1})}. \quad (2)$$

The correction step introduces the measurements into our estimates from the previous step, thereby improving them. $P(x_t|x_{t-1}, u_t)$ is the motion model of the robot. It estimates the probability distribution of the vehicle location on moving with motion command u_t. $P(z_t|x_t, m)$ is a measurement model which describes the probability of making an observation z_t given a location and a map of the surroundings. These steps can be implemented using a number of Bayesian filters which are explained in subsequent sub-sections.

3.1a *Kalman filter*: For any real life scenarios, both motion model and measurement model are non-linear in nature. Kalman Filter based approaches like EKF-SLAM (Guivant & Nebot 2001; Thrun *et al* 2006) linearise the motion model and the measurement model of the robot for prediction, and update equations to adapt the SLAM problem to EKF-filter equations. This linearization is carried out using first order Taylor's series expansion. As new terrain is explored, the number of features/landmarks in the environment increases. This results in an increase in the size of the state vector and the covariance matrix. Since EKF equations involve matrix operations including matrix inversion, the increase in the size of the covariance matrix leads to a prohibitive increase in the computation cost. The cost of updating a full stochastic map of N landmarks is $O(N^2)$ (Motalier & Chatila 1989). This makes such approaches unemployable for online operations for large areas. The other major problem is filter inconsistency. In particular, when the robot's orientation uncertainty is large, the extent of inconsistency is significant. This is because of imperfect linearization of the robot-motion model and the robot-measurement model (Castellanos & Neira 2004; Bailey *et al* 2006). As the filter diverges, uncertainty in estimations increases, and this results in improper loop closures. EKF-SLAM is also very fragile to incorrect data associations as there is no provision in EKF-SLAM to undo any committed data association. EKF-SLAM uses maximum likelihood data association. An incorrect data association leads to certain failure. EKF-SLAM approaches assume Gaussian noise models, which may not be true. Another problem with EKF is its inability to incorporate negative information (Thrun 2002a). For example, absence of an expected observation of a landmark represents the negative information. The use of negative information avoids false positives in data association.

MonoSLAM (Davison *et al* 2007) algorithm presents a method for doing SLAM using a monocular camera at real time (30Hz). This algorithm works best for room size feature rich environments. MonoSLAM generates a probabilistic feature based map of current estimates of feature locations and camera position using Extended Kalman Filter. The algorithm is aided by initializing it with the position of a known target. This helps MonoSLAM in estimating the correct scale of operation as a single camera cannot estimate depth information. As the camera moves, active search for features is carried out for existing features and new features. The state vector is updated accordingly. An open-source implementation of this work is available on the

author's website (Davison). In Strassdat *et al* (2012), authors have provided an analysis of filtering methods for visual SLAM.

One cause of error in EKF based solutions is linearization via Taylor's series expansion. Unscented Kalman Filter (UKF) (Thrun *et al* 2006) overcomes this via use of unscented transform, resulting in an equal or better result than provided by EKF based solutions. A set of points, called sigma points, are selected deterministically. These points are passed through non-linear motion and measurement models and the resulting mapping of these points is represented as a Gaussian. The UKF algorithm calculates without the use of any Jacobian, hence it is also known as a derivative free method. The unscented transform is equivalent to computing the first two terms in Taylor's expansion while EKF uses only the first term in Taylor's expansion. Thus, UKF provides better results for non-linear systems, but is slightly slower than EKF. UKF is much more suited for estimation of approximate Gaussian probability distribution (Thrun *et al* 2006).

3.1b *Information filters*: The efforts to reduce increased complexity either focus on (i) achieving approximate/sub-optimal solutions (Thrun *et al* 2004; Walter *et al* 2007) or (ii) they result in exact maps with a large reduction in cost as a result of delays in the global map updations (Guivant & Nebot 2001; Tardos *et al* 2002; Castellanos *et al* 2007), but still are of $O(n^2)$ complexity.

The information filter based method falls in the first category. The information filter (IF) is the dual of the Kalman filter (Thrun *et al* 2006). IF represents the Gaussian in a canonical representation *i.e.* using an information matrix Ω and an information vector ξ. Let μ and Σ be the mean vector and covariance matrix of the Gaussian distribution. The following relation holds

$$\Omega = \Sigma^{-1} \text{ and } \mu = \Sigma.\xi = \Omega^{-1}.\xi. \tag{3}$$

IF has much simpler update steps because of the canonical representation. The information matrix represents constraints between relative locations of a pair of features in the map. The strength of each link is related to the distance of the corresponding features. The more distance apart two features are, weaker is the link (Thrun *et al* 2006). Most features are connected to a small number of other features. Thus, most of the entries in the information matrix are relatively very small *i.e.* very close to zero, which can be neglected. The resulting approximation is a sparse information matrix. Memory required for storing a sparse information matrix is linear in the number of features. Because of the sparse nature of the information matrix, by use of efficient data structures and algorithms, computation efficiencies can be improved. Sparse Extended Information Filter (SEIF) based algorithms (Thrun *et al* 2004; Walter *et al* 2005) reduce the complexity of the algorithm using the key idea that maintains the sparsity of the information matrix by removing the weak robot-landmark constraints. SEIF provides efficient but approximate or suboptimal solutions.

All landmarks can be classified as either *active*, denoted as, m^+ or *passive*, denoted as, m^-. The active landmarks are the ones which are currently being observed by the robot. During filter update via motion model, the filter introduces new links between the active landmarks and weak links between the robot pose and the active landmarks. SEIF (Thrun *et al* 2004; Eustice *et al* 2005) maintains the sparsity in the information matrix by removing weak links between the robot pose and active landmarks. This can be achieved by reducing the number of active landmarks at any time. As a result, a sparsification sub-step is executed in the filter update step. The sparsification step removes the links between the robot's pose and the subset of active landmarks, denoted as, m^0 which become passive in the sparsification step. This sparsification is equivalent to a conditional independence of the robot's pose from the m^0 landmarks given the active (m^+)

and passive (m^-) landmarks. The time required for all the update steps is independent of the size of the map for known data association and logarithmic for unknown data association (Thrun *et al* 2006. Thus, SEIF is computationally efficient and can be employed as an online algorithm. The extraction of the mean vector μ is performed using efficient optimization techniques *e.g.* coordinate descent. It is faster than the EKF based solution. The quality of the map depends on the approximation involved in the sparsification and how accurately μ is computed. Improper sparsification could lead to filter divergence and to incorrect data associations.

Decouple SLAM (D-SLAM) (Wang *et al* 2007a) reduces the computational burden of filtering approaches by decoupling mapping and localization tasks of SLAM. SLAM problem is reformulated as consisting to two concurrent but separate processes: (i) non-linear static estimation problem for mapping and (ii) low-dimensional dynamic estimation problem for localization. The state vector is transformed and consists of two parts: part p_1 consists of distances and angles among the features, part p_2 consists of distance and angles between features and the robot pose. D-SLAM assumes that at each observation, the robot observes atleast two landmarks f_1 and f_2 which have previously been observed as well. All distances to observed landmarks are computed from this previously observed first feature f_1. All angles are measured *w.r.t.* f_1 and f_2. This way, all observations are represented relative to two previously observed landmarks. The mapping process is formulated using Extended Information Filter. For large environments, a significant portion of the environment remains unobserved. Thus, the information matrix largely remains sparse. Localization is achieved by combining the EKF-based local SLAM approach with the map input from the mapping task of D-SLAM. The information matrix in D-SLAM remains sparse. This eliminates the need of the sparsification step as in SEIF (Thrun *et al* 2004). The recovery of feature location estimates and respective covariances is performed using Preconditioned Conjugate Gradient method. The feature estimates can be achieved in $O(N)$ time, where N is the number of features. Since information about robot poses and their relation with features is not used in the mapping task, this leads to information loss in mapping. An extension to D-SLAM (Wang *et al* 2006) uses absolute feature location rather than relative location.

Exactly Sparse Extended Information Filter (ESEIF) (Walter *et al* 2005) maintains robot pose as part of the state vector. ESEIF is different from D-SLAM in the manner in which the sparsity of the information matrix is achieved. In D-SLAM, sparsity is achieved as a consequence of formulation of a separate mapping task. ESEIF imposes sparsity by performing updates on the information matrix by periodic solving for 'kidnapping robot and relocating problem'. In the robot kidnapping problem, the robot is picked from one place and placed in some other place. During this movement, the robot is not able to observe the environment. Hence, the robot is not aware of its relocation. Such scenario arises in some operations, where the robot is switched off at the current place and switched on again at some other place, by physically moving the robot, for restarting the operations.

3.1c *Particle filters*: All above approaches assume the probability distribution to be unimodal Guassian. Particle Filter (PF) (Doucet *et al* 2001) based approaches overcome the limitation of representation for non-Gaussian probability distribution characterized by data association problems. PFs are non-parametric recursive Bayes filters. A particle filter represents a probability distribution as a collection of weighted random samples. This allows PF to represent any arbitrary distribution. Different particles can represent different data association hypotheses. A Particle filter recursively performs the following three steps:

(i) Sampling: In this step, the filter generates a set of samples, called particles, to represent the distribution from a proposal distribution. Each particle represents a candidate solution.

More the proposal distribution is near to the actual distribution, earlier the filter converges *i.e.* the filter reaches a correct stable state.

(ii) Weighting: An importance weight is assigned to each particle. A particle with higher weight represents a better quality solution.

(iii) Resampling: This step selects a subset of samples according to the individual weight of the sample. Low weight particles represent unlikely or low quality solutions. Such particles are liable for rejection or replacement by higher weight samples in this step. It guides the filter to an optimal representation of the distribution.

PF are examples of a sequential Markov Chain Monte Carlo sampling approach, namely, sequential importance resampling.

Particle filter (Thrun 2002a) based SLAM approaches try to examine a huge number of hypotheses as samples of a probability distribution over the space of maps. Each particle represents a hypothesis. This allows for representation on non-Gaussian probability distribution. These approaches may not always identify a unique correct solution and might end up with an approximation of the environment. Particle filter based approaches do not make any assumption about the nature of noise in the system and the environment. The Particle filter based approaches require $O(MN)$ time, where M is the number of particle and N is the number of landmarks. Negative information can be incorporated by reducing the weight of the sample. Hence, these are better than Kalman Filter based approaches that assume the noise to be Gaussian.

Fast-SLAM (Montemerlo *et al* 2002; Montremerlo *et al* 2003) uses the Rao-Blackwellised particle filter (Doucet *et al* 2000) approach. The assumption is that given the correct location of the robot poses, measurements are independent of each other. SLAM problem is factored as a localization problem and N independent landmark identification problem (Murphy 1999).

$$P(X_{1:t}, m|Z_{0:t}, U_{0:t}, x_0) = P(X_{1:t}|Z_{0_t}, U_{0:t}, x_0).P(m|X_{0:t}, Z_{1:t}). \tag{4}$$

In FastSLAM, the robot trajectory is represented as weighted samples and the position of each landmark is represented as an independent Gaussian. While sequential important sampling is used for estimating robot poses, an EKF based approach is used for updation of each landmark given the robot pose. FastSLAM maintains a tree for data association, hence reducing complexity to $O(M\ \log(N))$.

The problem with particle filter based approaches is how to maintain diversity (after resampling) in minimal size set of particles to cover the complete probability distribution. As the robot explores a larger environment, more number of particles are required. This lead to an increase in computational load. The challenge is to reduce the number of particles required. In Grisetti *et al* (2007) & Stachniss (2009), authors provide an empirical measure to determine when to resample. This measure is based on weights of the samples. The intuition is that as samples become more true to target distribution, individual weights of samples tend to become equal. Thus, whenever difference in the weight of the samples varies over a threshold, a resampling step is performed. For grid maps, an improved proposal distribution is also explained. For accurate sensors, *e.g.* LIDAR, observation likelihood is peaked. Usually the motion model is used as proposal distribution. If the motion model is used as proposal distribution, its product with the observation likelihood will also be peaked. For covering a peaked distribution, a dense spread of particles is required. This will increase the computational burden. To reduce the number of required particles, an adaptive proposal distribution is centred around the maximum of likelihood function of the scan-match procedure. A scan-match procedure matches the current scan with another scan in the history to estimate the overlap between these two scans. A larger overlap is indicated by

a higher value of the maximum likelihood function. This method uses a Gaussian approximated around the maximum likelihood region to draw samples. This method is suitable to generate occupancy grid maps. Dieter Fox (Fox 2003) has presented an approach to adaptively bound the number of particles required through KLD (Kullback-Leibler Divergence)-sampling. This approach adapts the sample set size by computing KLD error between the sampled distribution and a discrete distribution computed over the complete map. An open-source implementation of Rao-Blackwellised Particle Filter based SLAM algorithm is available as a software package, named GMapping, with the Robot Operating System (ROS) (ROS 2014). It has been extensively used for generating occupancy grid maps for medium size environments.

Distributed Particle SLAM (DP-SLAM) (Eliazar & and Parr 2003) is a particle filter based SLAM algorithm for use with LIDAR sensor. DP-SLAM does not make any assumption about the landmarks. It uses particle filters to represent both the robot pose and the possible map. For efficient storage and maintenance of grid maps, only one grid map is maintained for all particles. Data structure used for storage of this single occupancy grid map is a matrix of grid cells. Each cell is represented as a balanced tree, called as the observation tree. This storage approach enables saving in memory requirements and time saving by avoiding copying of maps during the resampling phase. Each particle is assigned a unique identification number, called as ID. Whenever a particle makes an observation about a cell, it inserts its ID and the observation data into the associated tree. Each particle behaves as if it maintains its own private map. An ancestry tree is used to maintain the parentage history of each particle. All current particles are leaves. Parent nodes in the ancestry tree are particles from the previous iteration. Updates to the map are done using the ancestry tree. This ancestry tree allows the algorithm to maintain a consistent grid map. DP-SLAM2.0 (Eliazar & Parr 2004) provides an improvement in storing the probability of cell occupancy. DP-SLAM is an offline algorithm. The source code of this algorithm is shared by the authors as website (Parr). Figure 2b is generated using this code.

tinySLAM (Steux & Hamzaoui 2010) is a simple particle filter based algorithm which maintains a single map of the environment. It integrates the laser measurements into a map using a particle filter based localization in the environment. It is very easy to implement but its applicability is limited to small lab environments.

Range-only SLAM (RO-SLAM (Blanco *et al* 2008b)) is a probabilistic mapping approach where the evironment is observable only via beacons. The location of each beacom is estimated without any prior known information. A beacon could be an underwater beacon or wifi emitter. In this approach, authors have assumed that the observing sensor is able to distinguish between different beacons. Hence, the data-association problem is not present. The probability distribution of the location of each beacon is modelled as a sum of Gaussians. This probabilistic approach provides a much better estimation than a unimodal representation. The inference is performed using the Rao-Blackwellised Particle Filter. The location of each beacon is conditionally independent of the location of any other beacon. This observation was used in formulation of the inference algorithm. An open-source implementation of this algorithm is available at Mobile Robotics Programming Toolkit (MRPT) (Blanco 2014).

OctoMap (Hornung *et al* 2013; Wurm *et al* 2010) is a 3D map generating algorithm which uses an Octree representation for evolution of the map. An Octree is a hierarchical data structure for spatial subdivision of space in three dimensions. Each node represents a cubic volume, usually called a voxel. Each voxel could be further divided into eight equal size sub-volumes. A node in a octree is further divided, only if sensor data is available. Hence, this representation scales well in any direction, and allows representation in full 3D. Each volume can represent any property *e.g.* for a robotics application the representative property would be, occupancy of that volume. The occupancy of voxels is modelled probabilistically. As each voxel could have

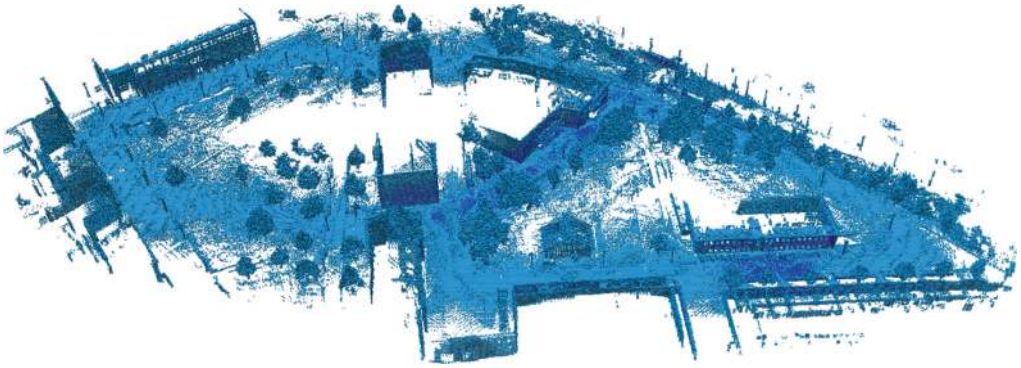

Figure 6. Octomap (Hornung *et al* 2013; Wurm *et al* 2010) output for Frieberg Campus dataset (Wurm *et al* 2010).

eight further divisions of its volume, it allows multi-resolution planning. This algorithm requires a good estimate of the robot's odometry for correct output of the map. Figure 6 shows the output of OctoMap algorithm on Frieberg-Campus dataset. This is generated using dataset and software available at author's website (Wurm *et al* 2010). OctoMap algorithm maps both free cells and occupied cells. In Figure 6, only the occupied cells are visualised. OctoMap algorithm is also available as a ROS (ROS 2014) package for manipulation of the objects by a PR2 (ROS 2014) robot.

Metrical map generating approaches discussed till now provide reliable localization in the frame of reference of the local neighbourhoods. However, these solutions have not been able to scale well to large-scale spaces because of computation costs and increased uncertainties in estimation of robot pose and location of landmarks. Odometric errors add up as the robot moves along long paths. Errors become more pronounced in environments with long paths with less distinguished features and in complex environments. In particular, these methods fail when the environment involves large loop closures. Feature Appearance Based Mapping (FAB-MAP) (Cummins & Newman 2008) and approaches discussed in section 3.2 have overcome these problems.

FAB-MAP (Cummins & Newman 2008) uses a probabilistic framework for a fast appearance based mapping which has been able to overcome these problems. FAB-MAP uses visual vocabulary *i.e.* images which encapsulate the visual features that are common in a particular type of environment. It uses a bag of words (image vocabulary) approach in which positive or negative observation of visual words in a scene is used to distinguish between a place already visited and a new place. A generative model which captures the fact that certain combinations of appearance words tend to co-occur, is learnt. Intuition is used to recognize the common objects in the scene as they lead to common appearance words. FAB-MAP 2.0 (Cummins & Newman 2009) has been demonstrated over a dataset that extends over a distance of about 1000 Km.

3.2 *Graph-based SLAM approaches*

The Graph-based SLAM (Thrun & Montemerlo 2006) approaches generate a network or graph of constraints from the measurements. In this approach, information about the environment is encoded as a graph of soft constraints. To obtain a refined map of the environment, these set of constraints are globally optimized so that errors due to constraints are minimized. Lu & Milios (1997) were the first to introduce this approach.

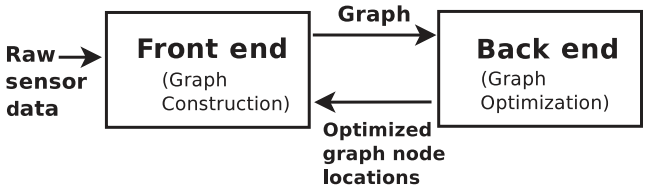

Figure 7. Components of Graph-based SLAM approach (Sünderhauf 2012; Konolige *et al* 2010). SLAM Front-end creates an abstract graph from sensor observations. SLAM Back-end optimizes the graph and returns graph node locations.

In Graph-based SLAM approach, the problem is divided into two parts: (i) SLAM Front-end: It is responsible for graph construction and (ii) SLAM Back-end: It is responsible for graph optimization. Figure 7 shows the interaction between front-end and back-end. During execution of the algorithm, front-end and back-end interact with each other. For online operations, these two sub-problems need to be solved alternatively.

Section 3.2a presents a detailed analysis of SLAM front-end systems. Section 3.2b presents a survey on SLAM back-end algorithms. Improvements to SLAM back-end systems are discussed in 3.2c and 3.2d.

3.2a *SLAM front-end algorithms*: In SLAM front-end, a graph is constructed using the measurements made by the sensors on the robot and robot motion details. The nodes in the graphs are either robot poses or the landmarks in the map. An edge between two nodes represent the spatial constraints between the two nodes. A constraint is the measurement of one node from the position of the other node. The edge between two sequential robot poses x_j and x_{j+1} corresponds to a motion event. The spatial constraint for such edges is generated using odometry data. The odometry data may be obtained using the wheel movement encoders or using the visual odometry (Geiger *et al* 2011), *etc*. The edge between nodes for robot pose x_i and landmarks L_i in the environment is created by using measurements Z_i made by robot at pose x_i (Thrun *et al* 2006). The set of landmarks L_i are observed in measurements Z_i . The notation is the same as described in Section 3.1. For more information visual odometry, we refer readers to KITTI vision dataset's odometry evaluation webpage (Geiger *et al* 2014). This webpage provides a tabular comparision of 34 odometry computation algorithms.

The front-end is heavily sensor dependent as it involves sensor-data interpretation to add edges between nodes and landmarks. Data association is a challenging task and sensor specific solutions are implemented. Olson (2008) presented a method based on spectral clustering for outlier rejection during graph formulation. For effective data association, maximum likelihood solution, statistical test such as Joint Compatibility Branch and Bound (JCBB) (Niera & Tardòs 2001) and χ^2 test have also been used.

The front-end could result in a very big resultant graph. Since the computation time needed by the back-end non-linear optimization can grow as a cubic in the size of graph. Such large graph can be bottle-necks. To overcome this, the graph is converted to a pose-graph by eliminating nodes in the graph which correspond to the landmarks. A pose-graph is a graphical representation of the environment where all nodes in the graph refer to robot poses. The nodes, which correspond to landmarks in the environment, are eliminated either by marginalization (Frank Dellaert & Michael Kaess 2006; Thrun & Montemerlo 2006) or by directly matching the sensor measurements (Thrun *et al* 2006). This leads to addition of constraints between pose nodes in the graphs from which this landmark was observable (Thrun *et al* 2006) *e.g*. between robot pose x_i

and x_j, $|i - j| \neq 1$. The edge now represents a probability distribution over the relative location of the two poses, conditioned on their mutual measurements (Grisetti *et al* 2010a). Such constraints are also known as loop-closure constraints. Loop closure constraints are derived using various approaches which include laser scan matching (Olson 2009a) and appearance based methods (Kon 2009; Olson 2009b; Cadena *et al* 2012; Galvez-Lopez & Tardos 2012).

RGB-D SLAM (Endres *et al* 2012) provides a front-end using the depth sensor *e.g*. Microsoft Kinect. This algorithm generates a dense 3D model of the environment. Kinect sensor provides a RGB image and a depth map for the same environment. In this four step approach (i) First, visual features are extracted from images. Visual features may be SIFT, SURF or ORB, *etc*. (ii) These are matched with visual features from previous images. (iii) For the matched visual feature, depth value is read from the depth map. This provides a point-wise 3D correspondence between any two frames. The relative SE3 transformation between these two frames is obtained using the RANSAC algorithm. (iv) This processing between any two frames is encoded as a edge in the pose graph. Each node represents a frame *i.e.* a sensor location. The edge between these two nodes represents the relative transformation *i.e.* movement from one sensor location to next location. The loop closure edges are generated by comparing the current frame to 20 previous frame. These 20 frames consist of (i) three most recent frames and (ii) rest are uniform sampled from the remaining frames.

This algorithm uses g^2o (Kümmerle *et al* 2011) algorithm as a SLAM back-end algorithm, which will be explained in the next section. After the optimization by SLAM back-end algorithm, a global consistent trajectory of the sensor movement is obtained. It has been shown to be useful for small indoor environments. This algorithm cannot function in outdoor sunny environment because of the sensor involved. An open-source implementation of this algorithm is available at author's website and also as a software package for ROS (ROS 2014). OctoMap (Wurm *et al* 2010) is used for visualization of the algorithm's results.

In Kerl *et al* (2013), authors have extended the previous approach (Endres *et al* 2012) to achieve better accuracy by reducing photometric errors and depth errors in RGB-D sensor based SLAM. In order to avoid accumulation of drift errors, key frames are selected from the input

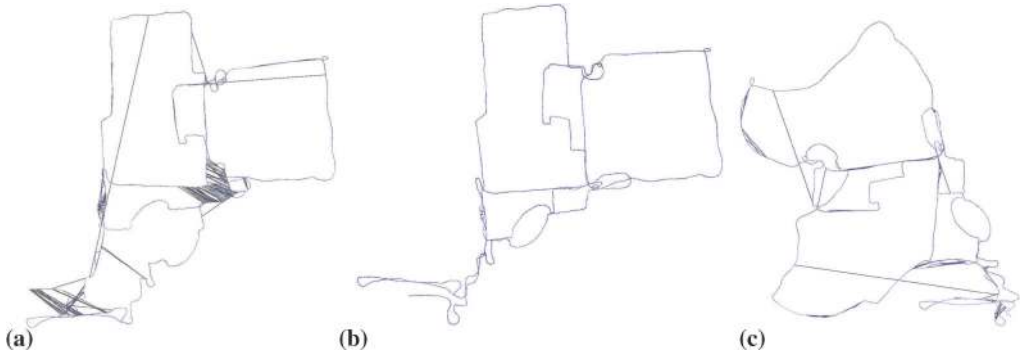

Figure 8. (**a**) The input dataset named Bovisa04. Bovisa04 dataset is available in RAWSEEDS (RAWSEEDS 2009). (**b**) The output of robust SLAM back-end algorithm RRR (Latif *et al* 2012a, 2013). It is able to reject wrong loop closures. The source code of RRR algorithm and input dataset are taken from author's website (Latif 2014). (**c**) Output of plain SLAM back-end. This output is generated using implementation of Gauss–Newton algorithm in g^2o (Kümmerle *et al* 2011) library. The source code of g^2o is available at author's website (Kuemmerle 2014).

stream of images. An entropy based method is implemented for this purpose. Incoming images are compared to the recent most key frame. On selection of a new key frame, 3D transformation *w.r.t.* previous key frame is computed using ICP. The loop closure constraints are computed between key frames. An open source implementation of these algorithms is available as a tool DVO (Kerl *et al*).

SLAM front-end approaches are not hundred percent accurate. A plain back-end SLAM algorithm will get misguided by these front-end errors and can result in inaccurate maps. Figure 8a shows a variant of Bovisa04 dataset in which false loop closures have been introduced. Figure 8c shows the output of a plain back-end algorithm. It is not able to identify the prescence of outliers in the output of SLAM front-end and results in a wrong map. Figure 8b shows the output of RRR, a robust SLAM back-end, algorithm. It detects the presence of wrong loop closures and do not considers such edges during the optimization. In section 3.2c, we will discuss robust back-end algorithms which can generate accurate results even in the presence of errors in the SLAM front-end output. Section 3.2b introduces SLAM back-end algorithms.

3.2b *SLAM back-end algorithms*: A SLAM front-end converts the sensor observations into an abstract graph. After construction of the graph, it is passed to the SLAM back-end. The back-end optimizes the robot poses at nodes such that the configuration of the robot poses maximally satisfy all the constraints *i.e.* the error introduced by the constraints is minimized and likelihood of observations is maximized. The back-end approach is agnostic to sensors as it operates on the abstract representation of the environment *i.e.* the graph. The constraints in the graph are non-linear because rotations are involved in robot motion (Huang *et al* 2010b). The constraints are linearized for purpose of optimization. During the non-linear optimization process, the posterior probability distribution of poses and landmark location given measurements and motion commands is converted into a system of linear equation. This is based on the assumption that the outcome of the robot motion and the measurements are normally distributed (Thrun *et al* 2006) and the system has zero mean Gaussian noise models. A tutorial on this approach is published by Giorgio *et al* (Grisetti *et al* 2010a). Once the correct robot poses are obtained, the map of the environment can be generated by appropriate placement measurements at corresponding points of the observations. The back-end implementation uses the properties of SLAM problems. The graphical model perspective for probabilistic inference is used. In the last decade, a large number of different efforts have been made to optimize the graph network. Some use Maximum Likelihood Estimation (MLE) using non-linear optimization methods such as Guass–Newton, Levenberg–Marquardt (LM) and sparse matrix factorization methods *e.g.* Cholesky, LDL and QR.

The SLAM back-end algorithms can operate in batch mode and online node. Graph-based SLAM algorithms which have also been extended for online-operations include SPA (Konolige *et al* 2010), TJTF (Paskin 2003), Treemap (Frese 2006), iSAM (Kaess *et al* 2008; Kaess 2008), iSAM2 (Kaess *et al* 2012), Max-Mixture (Olson & Agarwal 2013), RRR (Latif *et al* 2012a), SC (Sünderhauf & Protzel 2012), RISE (Rosen *et al* 2012), DDF-SAM (Cunningham *et al* 2010, 2013), COP-SLAM (Dubbelman & Browning 2013) and the algorithm in Kretzschmar & Stachniss (2012). Max-Mixture, RRR, SC and RISE algorithm are explained in Section 3.2c. DDF-SAM are explained in Section 3.4. The rest of the algorithms are explained below.

A number of approaches model the SLAM back-end problem using graphical models, such as Markov random graphs (Ranganathan *et al* 2007), factor graphs (Frank Dellaert & Michael Kaess 2006) and junction trees (Paskin 2003), so as achieve effective probabilistic inference.

Thrun & Montemerlo (2006) use a conjugate gradient based optimization method to find a maximum likelihood solution in 6D space. Olson (2008) explains the use of optimization algorithms to iteratively build a maximum likelihood map given a set of measurements. In Kümmerle *et al* (2011), the authors explain g^2o (general graph optimization), a general framework for graph optimization. The source code can be accessed at author's github repository (Kuemmerle 2014). g^2o provides algorithm implementation of various optimization algorithms with edge constraints from SE2 and SE3 groups. An experimental implementation of online and incremental graph optimization algorithms is also provided.

In Olson *et al* (2006), Edwin B Olson presents a method based on a variant of preconditioned stochastic gradient descent (SGD) to optimize the graph network. At each optimization iteration step, one constraint is randomly selected. The nodes of the network are moved such that error in the network due to this constraint is minimized. This way, all constraints are optimized one after the another.

Grisetti *et al* (2009) extended the SGD approach (Olson *et al* 2006), with the use of a tree based parametrization of the graph. This method is named as TORO. This resulted in better convergence speed and execution time. This method uses Spherical Linear Interpolation (Barrera *et al* 2004) for distributing rotational error. Translational errors are distributed after the rotation errors have been distributed to the nodes. Because of this, this method produced accurate 3D maps and maps for non-flat environments. To keep the number of constraints less, whenever the robot revisits a place, the current pose of the robot can be assigned to an existing node. The existing constraint for this node is updated with the constraint for latest observation. This keeps check on the number of constraints, and hence on computation time required for optimization of the graph. The source code of this algorithm is available at website www.openslam.org (Stachniss *et al* 2014).

Thin Junction Tree Filter (TJTF) by Paskin (2003) provides an incremental filtering solution based on junction trees. TJTF uses a lazy message passing scheme. New data is propagated to fixed near-by clusters, not to every cluster in the junction tree. The approximation is that the marginal of remaining clusters will not be conditioned on the observation (Paskin 2003). Or in adaptive message passing scheme, a message is passed only as long as the message induces significant changes in the belief state. The significance of a message from one cluster to another cluster over a separator is measured using the Kullback–Liebler divergence of the original separator marginal from the new separator marginal. This keeps the complexity of solving the inference on the junction tree manageable, resulting in linear-time filtering operations. An open source implementation of TJTF is available at website www.openslam.org (Stachniss *et al* 2014).

Treemap (Frese 2006; Frese & Schröder 2006) provides a highly efficient Gaussian inference mechanism based on a hierarchical decomposition of the region. The motivation of Treemap is to exploit locality of sensor perception horizon, as errors are small in local perceptions. Treemap expects the topology of the environment to be such that it can be recursively divided into halves with little overlaps. This decomposition leads to a hierarchy of regions which is stored as binary tree. Each region stores a set of only those landmarks which are observable from outside this region. To keep inference efficient, Treemap uses Kernighan–Lin heuristic (Hendrickson 1995) to keep the tree balanced. For computational gains, Treemap assumes that there are no common features between the regions. These sparsification methods work for assumed topological restriction. This approach is more effective for multilevel structures. The core computation of Treemap is similar to TJTF (Paskin 2003), but Treemap uses a different representation of the messages (distribution). The source code of this algorithm is available at website www.openslam.org (Stachniss *et al* 2014).

In most of the robotic operation, the range of the robot sensor is much smaller than the actual size of the environment in which the robot is deployed for the desired operations. In such deployment environments, the resulting graph of the front-end SLAM module is sparse in connection. This has lead to the use of sparse linear algebra methods in solution to SLAM. Frank Dellaert & Michael Kaess (2006) explain the use of factor graph (Kschischang *et al* 2001) to model the SLAM problem and the use of Bayesian inference methods to arrive at the solution. They also explain how inference on factor graph is equivalent to solving SLAM as least square problem. Frank Dellaert & Michael Kaess (2006) also explain the use of sparse matrix factorization for back-end optimization of the graph. The SLAM problem is formulated as a smoothing problem which is further transformed into a least square optimization problem. The matrix associated with smoothing is typically sparse. Using sparse linear algebra techniques such as Cholesky factorization and Levenberg-Marquardt algorithm, the smoothing information matrix is factorized into square root information matrix R. The matrix R is used to compute the optimal node positions. The authors also explain how good variable ordering techniques for variable elimination lead to reduce fill-in in the graph, resulting in significant saving in computation time. This provides an offline solution. An open source implementation of this algorithm is available as *GTSAM* (Georgio *et al*). Agarwal & Olson (2012) have presented a comparison of various variable ordering techniques. Ranganathan *et al* (2007) present a method which uses Loopy Belief Propogation (LBP) on Gaussian Markov random graph as generated in $\sqrt{SAM}$ (Frank Dellaert & Michael Kaess 2006). In Sparse Pose Adjustment (SPA) method (Konolige *et al* 2010), Kurt *et al* explain a method to extract the sparse graph matrix from the graph and the use of a variant of Levenberg–Marquardt non-linear optimizer to solve it in incremental and batch modes.

HOG-Man (Grisetti *et al* 2010b) uses a hierarchical pose graph representation of the graph for optimization. The output graph of the SLAM front-end is abstracted at various levels of detail. The lowest level is the original input graph and top level is coarsest, thus representing the problem as a hierarchy of graphs. The hierarchical pose graph is incrementally augmented with new observations at the lowest *i.e.* densest level. The change is propagated to the next higher level only if this could lead to a change in the graph at the next level. The optimization of the graph starts at coarsest *i.e.* highest level. The optimization operation is based on Gauss–Newton with Cholesky factorization on *manifold* based representation of the state space. A manifold is a mathematical space that is not necessarily Euclidean on a global scale but can be approximated as Euclidean in the local neighbourhood (Lee 2003). The manifold based representation avoids the singularities, which may occur during optimization due to non-Euclidean rotational components of the state space. The optimization results are propagated to the next lower level only if the optimization has lead to significant changes to the node configuration. This leads to gain in computation time. An open source implementation of HOG-Man is available at website www.openslam.org (Stachniss *et al* 2014).

The incremental Smoothing And Mapping (iSAM) (Kaess *et al* 2008; Kaess 2008) approach provides an incremental solution for the SLAM problem by performing fast incremental updates of the square root information matrix and efficient online data association. The incremental updates to the square root information matrix are carried out using Givens rotation (Golub & Loan 1996). iSAM is an improvement over $\sqrt{SAM}$ (Frank Dellaert & Michael Kaess 2006). This algorithm allows reuse of the most of previously computed square root information matrix. Linearization of measurement function and periodic variable reordering (Davis *et al* 2004) are performed in batch steps because these are computation intensive steps. These steps keep iSAM consistent and efficient for incremental operations. The source code of iSAM algorithm is available at author's website (Kaess 2014). Figure 9 shows the output of iSAM algorithm on Manhattan3500 dataset. This dataset is available with the source code of iSAM.

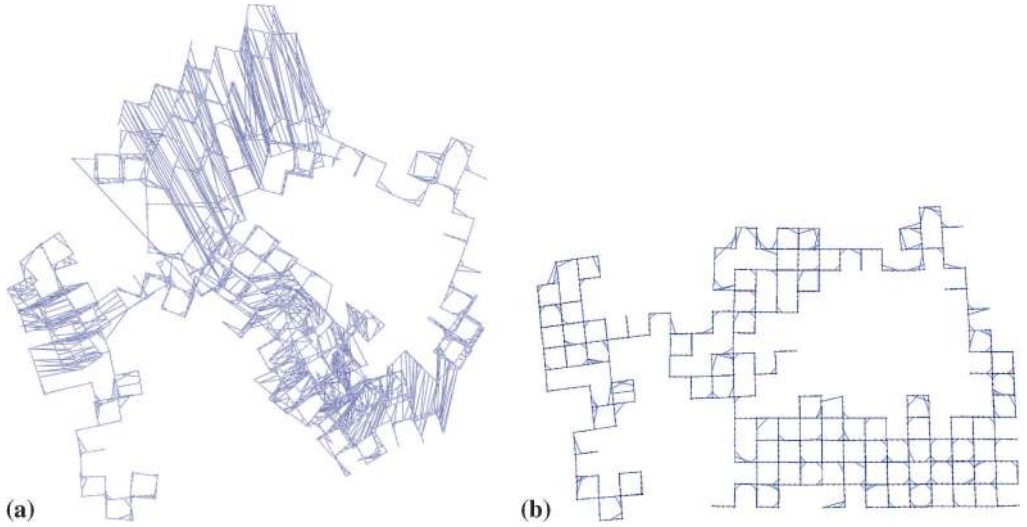

Figure 9. (**a**) Manhattan3500a is a simulated pose graph consisting of 3500 poses and 5598 constraints (including 2099 loop closing constraints). (**b**) The result of iSAM algorithm.

iSAM2 (Kaess *et al* 2012) has been made more efficient method by use of a novel data structure, the Bayes tree (Kaess *et al* 2010). A bayes tree is a directed graph where a node in the tree corresponds to cliques in the graph. It is similar to junction trees (Cowell *et al* 1999). The Square root of the Information matrix is stored as a Bayes tree. Any incremental measurement affects only a small part of the tree, as new measurements affect only a small subset of the complete state space. Only these parts of the tree are reverted back to the factor graph. By using a new variable elimination order, a new Bayes tree is found for this new graph. The unaffected parts of the tree are reattached. The variable reordering is done at every incremental update. This eliminates the need of periodic batch update step as in iSAM (Kaess *et al* 2008). iSAM2 allows fluidic relinearization of a reduced set of variables. iSAM2 relinearises a variable only if its estimate deviates from the linearization point more than some predefined threshold.

Rosen *et al* (2012) present an incremental method for inference on sparse factor graphs based on Powell's Dog-Leg optimization (Lourakis & Antonis 2005) algorithm. This method is called as Robust Incremental Learst-Squares Estimation (RISE) (Rosen *et al* 2012). This method is an improved version of iSAM. RISE is shown to be robust to non-linearity because of Powell's Dog-Leg otpimization. RISE method has performance comparable to iSAM. In Rosen *et al* (2013), the author further extend this approach by using non-Gaussian probability distributions for factorization.

3.2c *SLAM back-end: robust algorithms*: A lot of improvements to SLAM back-end processing have been presented in the SLAM literature. SLAM back-ends use least-square optimization methods. These methods are prone to outliers *e.g.* incorrect data-associations and false loop closure detection by the SLAM front-end. Such errors are common in real-world environments. A number of approaches overcome these problems by either allowing for better probability density modelling (Olson & Agarwal 2013; Rosen *et al* 2013; Pfingsthorn & Birk 2012) or by allowing inferencing mechanisms in optimization mechanisms (Sünderhauf & Protzel 2012; Latif *et al* 2012a; Rosen *et al* 2012). Using these two approaches, robustness to the SLAM back-end solutions can be obtained.

Sünderhauf & Protzel (2012) & Sünderhauf (2012) present the switchable constraint (SC) approach, to increase robustness of SLAM back-ends by subjecting the topology of factor graph to optimization. This is achieved by introducing a new variable, *switch variable*, in the factor graph for each constraint which could be an outlier. The most of outliers are because of wrong loop closure constraints. The incorrect loop closures are because of the inaccuracy of the SLAM front-end methods. SC is the first method to a provide mechanism to deal with SLAM front-end errors in SLAM back-end. A switch variable can have value on or off. If the value of a switch variable is *on*, the corresponding loop closure constraint is included in the optimization process otherwise not. The incorrect edges can be removed during the optimization process. This allows the topology of the factor graph to be subjected to optimization. The Switch variable influences the information matrix value of the corresponding factor component through a switch function $\Psi : \mathcal{R} \rightarrow [0, 1]$. The value can vary from original value to zero, thus influencing the covariance associated with the factor. These methods fail in some degenerate cases. Some environments consist of distinct parts, which are connected by sparse connections *e.g.* the Parking Garage dataset. In degenerate cases, if outliers are present in the sparse connection between distinct parts of the environment, this method fails. SC fails in such cases because of the insufficient amount of information connecting distinct parts of the environment. Dynamic Covariance Scaling (DCS) (Aggarwal *et al* 2013) proposes a closed form solution for computing value of the switch function for each individual loop closing constraint. DCS do not introduce switch variable as in SC. DCS computes a upper bound by which information matrix component for any loop closure constraint can be scaled. It implements a robust cost function (M-estimator). DCS results in much faster converges than SC to solution even in the presence of outliers. The source codes for SC and DCS are released by respective authors at website OpenSLAM (Stachniss

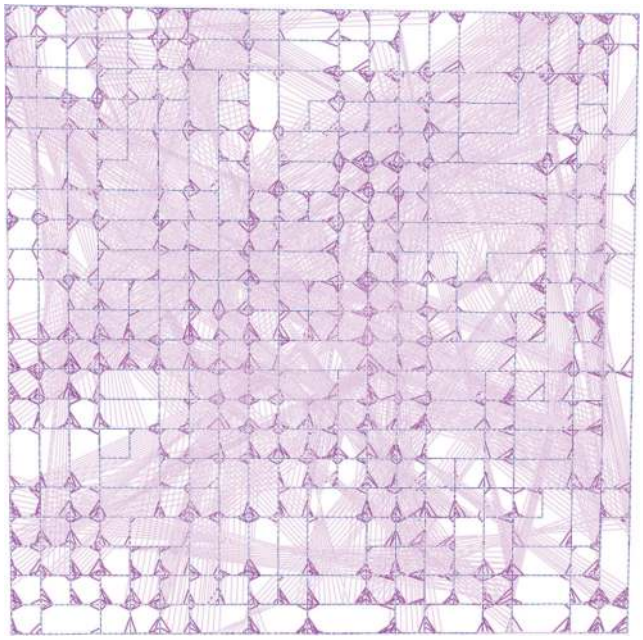

Figure 10. Figure depicts the output of City dataset of DCS algorithm. The edges in light pink colours have been rejected during optimization and dark edges have been retained.

et al 2014). Figure 10 shows the output of DCS algorithm on City dataset. The input dataset was infused with 0.1 std. dev. noise in edges.

The Realizing, Reversing, Recovering (RRR) (Latif *et al* 2013, 2012a) algorithm optimises the pose graph using the intuition that correct loop closure constraints agree with the odometry constraints and are consistent with each other, on the other hand wrong loop closures constraints are not. Thus correct loop closures in conjunction with odometry can be used to identify incorrect loop closures. This approach assumes that odometric errors are small. On optimizing the pose graph with wrong loop closure constraints, larger metrical changes will be required in comparison to optimization with correct loop closures. The false positive loop closure identification is performed using a number of χ^2 tests for checking the consistency of the loop closure constraints and deviation in resultant metric change from odometric data after optimization of the graph. This approach first clusters the loop closure constraints according to their timestamps. A threshold is used to define the neighbourhood. After each cluster formation, *intra-cluster consistency* check determines whether the complete cluster is consistent. If yes, this test checks for single constraints which are inconsistent in this cluster and rejects them. The test is performed as optimization task on a graph which contains all odometric constraints and loop closure constraints only from this cluster. The acceptance of constraints or clusters is based on χ^2 error tests. After this, in *inter cluster consistency*, the mutual consistency of clusters is determined. The aim of the algorithm is to find the biggest set of consistent clusters of loop closure constraints. During long-term experiments, new observations can be integrated session by session. This way previous possible failures can also be corrected. This approach is further extended for online operations in iRRR (Latif *et al* 2013, 2012b). An open source implementation of RRR algorithm is available at author's webpage (Latif 2014). Figure 8b shows the output of RRR algorithm on Bovisa04 outdoor dataset. This dataset is available with the author's released source code. Author has also provided a script to introduce errors in loop closure constraints.

In the conventional formulation of pose graph SLAM using factor graphs, it is assumed that all distributions are unimodal Gaussians (Sünderhauf & Protzel 2013). An ambiguous data association cannot be accurately represented using unimodal probability distribution. This representation is also not sufficient to model arbitary probabilistic distributions *e.g.* non-Gaussian noise. The source of such noise could be odometric errors (because of slip or skid) and wrong data associations (leading to incorrect loop closures). The odometric sensor is not able to capture the robot movement because of the slipping of wheels. It can happen because of the nature of the current operating surface. The odometric sensor is also not able to capture the non-movement of the robot in situations where the wheels are moving but the robot is not. This problem is known as the *Slip or Grip* problem. In the Max-Mixture (MM) Models (Olson & Agarwal 2013) approach, the spatial constraints are represented as a max-mixture of Gaussian distributions rather than a unimodal Gaussian. This approach permits representation of complex non-Gaussian probability density functions. The loop closures are represented using a two-component max-mixture (i) the front-end loop closure, (ii) a null hypothesis with a very low weight. The null hypothesis represents the possibility of a wrong loop closure. It is represented with the same mean as a first component and with a very large covariance. During optimization, a maximum likelihood selection process selects the best component. This component maximizes the likelihood of the factor's error function. To handle the Slip or Grip problem, the two component mixture model is proposed. The components are: (i) 15% noise model with large weight, (ii) mean centred component with low weight. To have an alternate solution for detection of the Slip or Grip other than odometry, a scan matching system is used to generate loop closures. This improved odometry model results in improved maps. The weights of the mixture are tunable parameters. The

insights of sparse factorization and variable order heuristics are applicable here as well for efficient computation. This method is demonstrated to operate in both modes: online and batch. In Pfingsthorn & Birk (2012), the authors have explored the use of mixture of Gaussian (MoG) to model probability distributions. But it has been shown that Max-Mixtures approach provides a computational advantage over the MoG (Olson & Agarwal 2013).

In Sünderhauf & Protzel (2013), authors present a comparision of SC, MM and RRR methods. Niko *et al* report that RRR performs best, followed by SC on real world datasets the Bovisa-06 (RAWSEEDS 2009) and the Bicocca (RAWSEEDS 2009). MM does not perform well on the real world dataset as it rejects a lot of correct loop closures. Whenever the error in the initial guess is very high, MM is prone to select wrong mixture components. On synthetic datasets Manhattan and City10000, additional false positive loop closures were added. SC and MM outperform RRR on synthetic datasets. RRR is not able to cope with false positive loop closures. SC has an advantage over the other two algorithms as it has only a single variable to tune.

3.2d *SLAM back-end: compressed graph algorithms*: The computation time and memory used by the SLAM back-end algorithms is directly influenced by the size of the input graph to optimize. Most of the current approaches use only the robot pose graph for optimization. Some approaches prune the graph. Cyrill *et al* (Kretzschmar & Stachniss 2012) presents information theoretic compression of the pose graph. Based on the analysis of the information gain achieved by addition of the input laser measurement, this measurement is either rejected or added. This compression method is lossy. The marginalization of a node could lead to fill-in edges in the graph, which destroys the sparsity of graph structure. To avoid this, an approximate marginalization scheme is introduced. Whenever a node is marginalized, it introduces a clique in the graph. This elimination clique is replaced with a tree-shaped approximation *e.g.* the Chow–Liu tree (Chow & Liu 1968).

Size of the graph to be optimized can also be reduced by converting the pose-graph to a pose-chain graph. The pose-chain graph contains the nodes only representing robot poses, not those representing landmarks as in the pose graph. In this sparse graph, pose nodes are strictly ordered in time for edge connections. This can be understood with the help of figure 11. Pose-chain graph contains only two kinds of edges (Dubbelman & Browning 2013) (i) Successive edges: which connect successive nodes *i.e.* x_t and x_{t+1}. (ii) Loop closing edges: these edges connects nodes for time step t back to time step $t - l$. Each edge represents relative pose displacements. Pose-chain graphs are generated by the SLAM front-end systems. Such graphs are natural outputs of SLAM front-ends which use accurate and reliable data-association methods. Such front-ends have low error due to drifts. In Closed-form Online Pose-chain SLAM (COP-SLAM) (Dubbelman & Browning 2013), multi-loop pose-chain graphs are optimised using trajectory bending (Dubbelman *et al* 2010). In trajectory bending, piece-wise distribution of displacement is performed over the

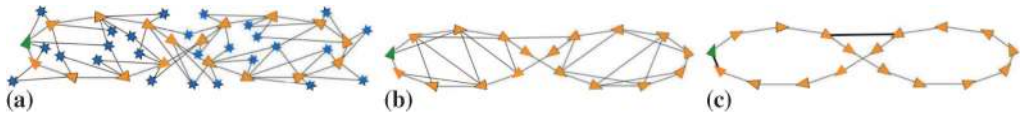

Figure 11. Different types of graphs in SLAM (Dubbelman & Browning 2013). (**a**) represents a general slam graph. (**b**) represents a pose graph. (**c**) represents a pose-chain graph. In this figure, triangles represent the robot poses. The initial robot pose is shown in green color and subsequent absolute robot poses are shown in orange colour. Landmarks are shown as blue stars. Edges model the landmark observations and relative pose displacements. In (**c**), dark edges represent the loop closure hypotheses.

complete trajectory. COP-SLAM is computationally more efficient than g^2o (Kümmerle *et al* 2011) but less accurate. The reason is that COP-SLAM is only piece-wise optimal for SE(2) and SE(3) for trajectory bending. The second reason is that COP-SLAM does not back propagate the optimization information of new loop closures to previous loops. The SLAM front-end of COP-SLAM uses visual odometery and appearance based methods for data association.

3.3 *Submap joining SLAM approaches*

A number of approaches aim to ease the complexity of the SLAM problem by joining a number of submaps. A submap is a metrical map of a small size. Most approaches keep the size of the submap small enough to avoid loop-closure problems. As the metrical map generating algorithms are efficient and accurate for small size maps. Each submap is an accurate description. Submap joining approaches attempt to achieve efficient and accurate results by joining these tiles of small size submaps. It has been shown that nonlinearity and non-convexvity is reduced by using submap-joining approaches (Huang *et al* 2010b). These approaches include Hierarchical SLAM (Estrada *et al* 2005), Divide and Conquer SLAM (Paz *et al* 2008), Techtonic-SAM (T-SAM) (Ni *et al* 2007), SLSJF (Huang *et al* 2008b; Huang *et al* 2008a), 3D-I-SLSJF (Hu *et al* 2009) and Linear SLAM (Shao *et al* 2013). Here SLSJF stands for Sparse Local Submap Joining Filter.

Hierarchical SLAM (Estrada *et al* 2005) and Divide and Conquer SLAM (Paz *et al* 2008) are based on the intuition. Hierarchical SLAM builds an adjacency graph of the independent local maps, using the relative position of the local maps. To build a global map, it combines the local maps in the order of their generation. Here, non-linearities have been successfully addressed using standard non-linear least-squares optimization techniques (Deans & Hebert 2000; Newman & Leonard 2003).

Another application of this approach is Monocular SLAM (Clemente *et al* 2007). This is a vision sensor based algorithm. It uses a web camera as the only sensor because of which it cannot provide any scale for the map.

Techtonic-SAM (T-SAM) (Ni *et al* 2007) is a divide and conquer approach using graph-based SLAM formulation. This algorithm is offline in nature. T-SAM does not differentiate between pose nodes and landmark nodes. It partitions the graph into submaps using a k-way cut mechanism. Each submap is optimized independently as in $\sqrt{SAM}$ (Frank Dellaert & Michael Kaess 2006). Each submap maintains a local coordinate system and local base pose. Some measurements span the submaps *i.e.* these are involved in more than one submap, called *inter-measurements*. The nodes corresponding to such measurements are called *separators*. The complete set of submaps are aligned together by optimizing the separator nodes. In smoothing and mapping approaches, one of the most computationally expensive operation is linearization for obtaining the measurements (Frank Dellaert & Michael Kaess 2006). T-SAM caches the linearization points and the factorization of the submaps. Since each submap uses a local coordinate system, the linearization point remains valid even after the local base pose has undergone transformation. While aligning the submaps into a complete map, only the nodes corresponding to inter-measurements are required to be linearized. The cached factorization of the submaps is also reused for gain in computational time. This approach provides a better result compared to the EKF-based approaches as it is exact in nature. T-SAM can be used to solve large-scale problems which cannot be solved in one go due to limitations of computer memory as T-SAM needs only one submap and one separator in memory at a time.

Sparse Local Submap Joining Filter (SLSJF) (Huang *et al* 2008b; Huang *et al* 2008a) is local submap joining algorithm for building large-scale feature based maps. A local map may be

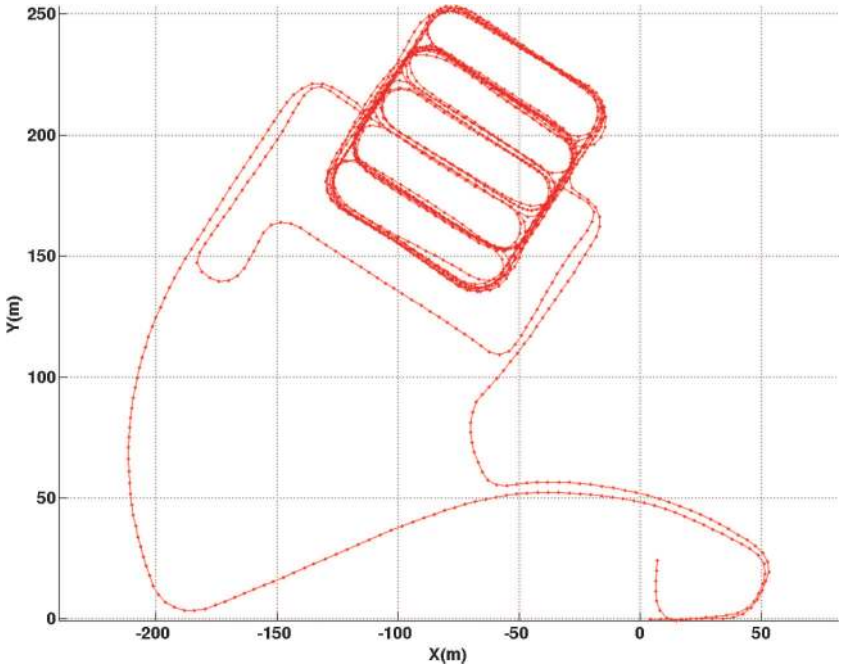

Figure 12. The output of LinearSLAM algorithm for 3D Parking Garage dataset.

generated using any reliable SLAM algorithm. SLSJF fuses each local submap into a global map one at a time. SLSJF uses Extended Information Filter to exploit the sparsity of information matrix for computational gains. In Hu *et al* (2009), authors further extend this approach to handle 3D local submaps.

LinearSLAM (Shao *et al* 2013) makes the assumption that the two submaps for joining should be in the same coordinate frame. The map joining operation requires solving linear least squares optimization problem, followed by non-linear coordinate transformation. The strategy to combine submaps may be divide-and-conquer or sequential map joining, but two submaps should be in the same coordinate frame. An open-source implementation of this algorithm is available at author's webpage and at website www.openslam.org (Stachniss *et al* 2014). Figure 12 shows the output of Matlab implementation of LinearSLAM algorithm for 3D Parking Garage Dataset. This dataset is available with the source code as a sequence of small submaps.

3.4 *Metric SLAM using multiple robots*

For a number of application *e.g*. exploration and search, a team of robots provides a better solutions. Most of the multi-robot mapping strategies are extension of the single robot SLAM approaches *e.g.* Kalman filter, Particle Filter, $\sqrt{SAM}$ and its extentions. The challenge for multi-robot SLAM includes when and how robots share their map with other robots.

Particle Filter based methods have been demonstrated in Howard (2006) for SLAM using a team of robots. In Howard (2006), the author presents an online mapping algorithm. In this approach, each robot is capable of generating maps using Rao-Blackwellised Particle Filter methods. It is assumed that during the life-time of the operation, each robot will atleast once observe other robots. It is also assumed that robots have the capability to identify other robots.

This mutual observation of robots allows the computation of their pose relative to each other. Consequent to determination of relative poses on the first encounter, the observations made by the observed robot are included into the map of the observing robot, by assuming that the observed robot is driving backwards. The inclusion of the recorded observation into the current map of the robot is achieved by feeding recoded observations to the filter in reverse time-order.

In Andrew Howard & Matarić (2006), authors present a SLAM approach for two dimensional space using manifold representation. The authors assume that data from all robots is sent to a central location and processed in this central location. In this approach, the Maximum Likelihood Estimation (MLE) technique is adapted for manifold representation. The advantage of manifold representation based approach lies in its delayed data association capabilities. The manifold is discretized into a set of overlapping patches. Each patch has a local planar coordinate system and finite extent. A set of pose constraint relations are defined between patches using scan matching algorithms and odometry. A local map is a set of patches such that all patches fit together. The best fit set is obtained using the MLE approach. The mutual observation of the robot is also used to establish a correspondence between the points on the manifold, which leads to loop closures. During loop closure operations, the robot may be required to halt because of computational delays.

A map fusion approach based on neural-networks is presented in Saeedi *et al* (2011). Each robot generates a occupancy grid map using EKF and a laser sensor. These occupancy grid maps are fused together to generate a compound consistent map. Each occupancy grid map from each robot is preprocessed using segmentation. Each segment is fed to a Self Organizing Map (SOM) (Ultsch & Siemon 1990) algorithm for clustering. Since the number of clusters is much smaller than the number of cells in the map, the processing of clusters requires much less time during map fusion. To account for uncertainty of occupancy grid map cells, cells with higher probability of being obstacles are used more frequently as inputs for SOM training. The map of the world is abstracted as arrangement of cluster points by the SOM. To obtain relative orientation between clusters, the Radon transformation (Deans 1983) is used. The relative translation between clusters is obtained using the norm vector of cluster points. This is followed by a verification step. A sum of squared-euclidean-distance-metric is used to measure accuracy of the match between clusters. A verification index is used to measure the similarity of maps. The purpose of the verification step is to choose the best hypothesis among candidates or reject false matches.

The D-SLAM (Wang *et al* 2007a) framework is also used for multi-robot SLAM (Wang *et al* 2007b). The prior knowledge of the robots' starting locations is not required. Each robot generates local maps using EKF SLAM. These local maps are fused into a global map using the D-SLAM framework. Each local map is treated as a single local observation of the environment for fusing in the global map. For finding overlap between local maps, an approach based on robust point feature matching of medical image registration method is used. The joint compatibility test is used to select among the candidate hypotheses.

A number of approaches are based on graph-based SLAM. These include C-SAM (Andersson & Nygards (Andersson & Nygards 2008), DDF-SAM (Cunningham *et al* 2010, 2013), (Reid & Braünl 2011), (Olson *et al* 2013), *etc*. Collaborative Smoothing and Mapping (C-SAM) (Andersson & Nygards 2008) is multi-robot robot SLAM based on $\sqrt{SAM}$ (Frank Dellaert & Michael Kaess 2006). Each robot generates a local map using $\sqrt{SAM}$. The robots share local maps with each other. The local maps joining is performed using rendezvous observations and features of the environment. The rendezvous observations refer to the observations when the robot observes other robots. A new set of constraints for these common observations, called connector, is added to the optimization process. For alignment of robot maps to a common reference

frame, a new node called base node is added. This base node is used to define nodes of one robot in the coordinate frame of other robot. A distributed mapping algorithm based on maximum likelihood estimation using Gauss-Seidel relaxation is presented in Rizzini & Caselli (2010).

DDF-SAM (Decentralised Data Fusion Smoothing And Mapping) (Cunningham *et al* 2010) is an extention of $\sqrt{SAM}$ (Frank Dellaert & Michael Kaess 2006) for multi-robot mapping. DDF-SAM achieves fusion information in a decentralised manner. Each robot in a team of robots is equipped with a SAM (Frank Dellaert & Michael Kaess 2006) module, a local optimization module, communication module and neighbourhood optimizer module. SAM module is used to generate local maps of the environment. A local optimization module compresses the local map into a condensed graph by marginalizing out all the pose nodes, as only landmark map is needed by the other robots. This helps in conserving communication bandwidth. Each robot maintains a neighbourhood, a bounded size set of robots in its communication range. This neighbourhood can vary with time. Each robot shares the condensed graph within its neighbourhood using the communication module. Each robot maintains a cache of such neighbourhood condensed graphs. The Neighbourhood Optimizer module merges the cached condensed graphs into a single neighbourhood graph. This is performed as a batch operation. Each robot maintains two incomplete maps of the environment: the local map and the neighbourhood map. This separation between local and neighbourhood map is maintained to avoid double counting of the information. Double counting of the information could lead to over-confidence in these observations. A Constrained Factor Graph (CFG) representation is used for generation of single neighbourhood graphs. CFG contains hard constraints to represent the frame of reference constraints and data association constraints between the neighbourhood graph and the local graph available with the robot. This leads to a constrained non-linear optimization problem. As each robot maintains a cache of neighbourhood maps, DDF SAM is robust to node failures and network topology.

DDF-SAM reconstructs a new neighbourhood map, whenever new summarized maps are available. This operation becomes increasingly costly with increase in area-coverage by the robot. DDF-SAM 2.0 (Cunningham *et al* 2013) provides this improvement over DDF-SAM (Cunningham *et al* 2010). Using DDF-SAM 2.0 approach, each robot maintains a single consistent map, *Augmented Local Graph*. The local and neighbourhood measurements are summarized into a single incremental Bayes Tree solver as in iSAM (Kaess *et al* 2008). To avoid double counting of information, *anti-factors* are used in the SLAM graph. An anti-factor is a factor which results in subtraction of information. Before sharing the map with other robots, *summarization* of the graph is done by introducing anti-factors for neighbourhood measurements *i.e.* those measurements which have come from other robots. Summarization can be performed by marginalization using Schur Complement Reordering for exact inference or using Naive–Bayes approximation. DDF-SAM and DDF-SAM2.0 approaches assume known data associations. Cunningham *et al* (2012) proposed a RANSAC based matching method. A Delaunay traigulation of the landmarks is computed. A set of correspondences C is calculated between two set of triangles using geometric features, for example, perimeter of the triangle and area of the triangle. The RANSAC algorithm is used on C to find the correct matches while avoiding ambiguities due to the outliers.

Reid & Braünl (2011) demonstrate a large scale multi-robot graph-based non-linear map optimization approach for a hybrid decentralized and distributed computation in Mutli Autonomous Ground-robotics International Challenge (MAGIC) 2010. In MAGIC 2010, the challenge for a team of robots was to map an urban semi-structured environment while at the same time avoiding static and dynamic dangerous elements in the environment. In this approach (Reid & Braünl 2011), each robot has a local SLAM module which generates submaps of a predefined size. Each submap has its own local coordinate system. Primary sensor used was the 2D laser. After closure

of a submap, it is compressed and broadcasted. The robot's local pose estimate is also broadcasted as odometric constraints between the broadcasted map and the new map. The Mapbuilder module is a SLAM back-end module and each robot has this module. It has three components (i) Submap Optimizer (ii) Submap Builder and (iii) Submap Matcher. The Submap Optimizer module is an implementation of SPA (Konolige *et al* 2010) for multi-robot computation. Each robot maintains its perception of the consistent global map. A Ground Control Station (GCS) is designated as *master* and all other ground robots as *slaves*. To have a consistent global map across all the robots, GCS periodically broadcasts the correction to the global map. All other robots incorporate these corrections in their map. This approach is called as hybrid-decentralised approach. Each robot also has a Submap Matcher module to generate spatial constraints between submaps in a local neighbourhood. This module looks for potential loop closures. Submap Matcher is an efficient implementation of scan matching (Olson 2009a) on GPU for parallel matching of submaps. On GCS, this module searches for matches which could enable large loop closures. Submap Builder module fuses all overlapping submaps into a single map using the GPU for visualization purposes.

Olson *et al* (2013) demonstrated a centralized mapping system for a team of 14 robots in the MAGIC 2010 competition. Each robot periodically generates a maplet of its surroundings and transmits it to a ground control station (GCS). The GCS fuses these maplets together by performing inference on a factor graph. Each maplet acts as a node in the graph. The edges in the graphs are generated using odometry, IMU and scan matching modules. The key idea implemented in scan matching modules is to use a multi-resolution matching system (Olson 2009a). In order to reduce the false positive matches, a loop validation module is implemented. The idea is that multiple matches agree with each other (Olson 2008). A human operator can override any match and force matches among some maplets to improve the quality of the map.

Lee *et al* (2012) present a survey on techniques for Cooperative-SLAM (C-SLAM).

A number of opensource metrical SLAM algorithm implementations are available on the OpenSLAM website (OpenSLAM 2014).

4. Qualitative map generating approaches

Qualitative map generating approaches build a topological map that describes the environment as a graphical representation having nodes as places and objects of interest, and edges as the spatial relation or path between the vertex. In this paper, the terms qualitative map and topological map are used interchangeably. Topological maps provide a more compact representation of the environment as compared to global metrical maps. This representation also allows for high level symbolic reasoning for map-building, navigation and planning. One of the problems in the metrical mapping process is accumulation of odometric errors over time. Qualitative map building approaches allow a map to evolve without worrying about the metrical aspects. Since the environment is abstracted to a graph in this representation, odometric errors that accumulate between graph nodes do not necessarily accumulate across the global frame of reference. This allows room for overcoming problems like loop closures, which are a result of metrical uncertainties. The idea is, that in order to solve complex loop closing problems in a tractable manner, a robot should be able to reason with a symbolic topological map. Also, the metrical map representations do not accurately represent the incompleteness in the information gathered by the robot during the exploration.

The problem with such approaches is how to achieve reliable abstraction of meaningful symbols from a continuous, noisy perception of the environment. In simple words, how can the robot

reliably detect, identify and recognize elements of the topological map, for example places and paths. In general, this is known as the *Symbol Grounding problem*. Some approaches use robot motion control as a way to detect and identify the distinct place (Choset & Burdick 1995; Kuipers *et al* 2000; Marinakis & Dudek 2010; Tao *et al* 2011). Other approaches make use of sensor data to extract unique landmarks or places (Ranganathan *et al* 2006; McClelland *et al* 2013; Paul & Newman 2010).

A number of qualitative approaches generate a Generalised Vornoi Graph (GVG) (Choset & Burdick 1995) as the output. A vornoi graph consists of nodes corresponding to positions in the map which are three way equidistant to the obstacles in the environment. The edges represent feasible paths such that each point on the edge is two way equidistant from the obstacles. Using various robot motion control laws, the robot is made to travel in a manner so as to follow the vornoi graph (Choset & Burdick 1995). The problem with this generalised vornoi graph is that the robot should always maintain sufficient number of obstacles in its sight. This is not achievable during exploration in open areas. The Saturated Generalized Voronoi Graph (S-GVG) (Tao *et al* 2011) overcomes this limitation by following a single obstacles at a saturated distance. The saturated distance is a predefined distance which shall be maintained from the obstacle, in case when only one obstacle is visible. The loop closing is performed using a multi-hypothesis filter. A hypothesis tree is constructed. Each level of tree corresponds to a different time-steps. The leaves in this tree represent the latest time-step. A posterior probability is assigned to each hypothesis. This value measures the fitness of the hypothesis to the observed sensor data. A low probability value hypothesis poorly fit the sensor observations. To achieve computational efficiency, the algorithm periodically prunes low probable hypotheses from the hypothesis tree.

Ranganathan *et al* (2006) present a method, named Probabilisitic Topological Maps (PTM), which uses Bayesian inference in the space of topological maps. The intuition is that all possible loop closure hypotheses will be considered during the process of evaluation of the hypotheses. Each hypothesis will represent a probable topology with a probability value. Low probable topologies will be rejected. Authors use appearance data in conjunction with odometry to assign a probability to each candidate topology. This provides a measure of correctness of the topology. The Fourier signatures of the panoramic camera images are used as appearance. The space of topological maps is very large because of the perceptual aliasing problem. A Markov-chain Monte Carlo (MCMC) sampling algorithm, the Metropolis–Hastings algorithm is used to approximate the posterior probability distribution over the space of candidate topological maps, given the sensor measurements. In this approach, for sampling purpose, each topology is modelled as a set partition over the set of all the landmarks. Each set in this set partition corresponds to a single physical landmark. The sampling algorithm samples over the space of these set partitions. Online Probabilistic Topological Maps (OPTM) (Ranganathan & Dellart 2011) approach extends the PTM approach for online operations. A Rao-Blackwellized Particle Filter (RBPF) is used to maintain joint posterior distribution of landmark locations and topologies. The landmark locations posteriors are computed analytically.

Spatial semantic hierarchy (SSH) (Kuipers *et al* 2000) is an influential work on topological map generation and navigation. SSH provides a framework in which large-scale spatial knowledge is represented over different ontological levels. It is inspired by cognitive findings about how humans assimilate and process large-scale spatial information. SSH describes a computational model that includes abduction of topological maps from the sensorimotor experience of a robot. As shown in figure 13, there are four levels of knowledge representation in SSH, structured in a hierarchy of abstraction and dependencies. As explained in Savelli (2005), at the *control* level, the agent repeatedly selects a hill-climbing control law to converge to and localize at a distinctive state ($dstate$); and then selects a trajectory-following control law to move from the

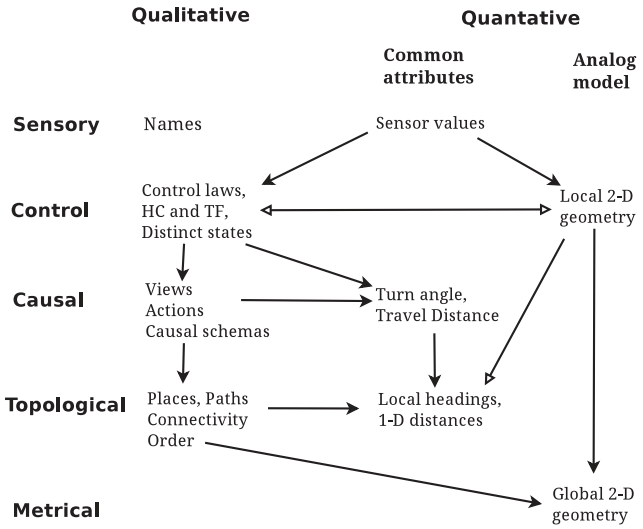

Figure 13. The Spatial semantic hierarchy (Kuipers *et al* 2000). Closed-headed arrows represent dependencies; open-headed arrows represent potential information flow without dependency.

current dstate to the neighbourhood of another. Thus hill-climbing converges to the next dstate, eliminating cumulative errors. This way, the controlled movements of the robot help to avoid and solve the correspondence problem. The *causal* level abstracts this pattern of behaviour to a deterministic automation, consisting of states (the distinctive ones), actions (sequences of control laws), $schemas < x, a, x' >$ (asserting that state x' results from performing action a in state x), and $views$ (the perceptual images of states, $view(x, v)$). The *topological* level distinguishes between turn and travel actions, and aggregates states into places, paths, and regions, related by connectivity, order, and containment. The *metrical* level consists of local metrical attributes for objects at the causal and topological levels, local metrical models of small-scale space in place neighbourhoods, and (when resources permit) global metrical models of the large-scale environment (Modayil *et al* 2004). A formalization of the topological map in non-monotonic logic, and an algorithm for identifying minimal models according to a prioritized circumscription policy is given in Remolina (2001). The circumscription policy formalizes preference and default criteria, which are well suited for the ontology of many real-world classes of environments such as offices or urban street networks.

In Marinakis & Dudek (2010), the author presents a solution for pure topological mapping. A graphical environment for robot operations is assumed, along with the robot's ability to detect landmarks and cyclic labelling of edges. To avoid a number of data association errors, the author presents a number of exploration strategies such that the robot revisits a number of potential problem prone areas. As the robot explores the environment, the candidate topological hypotheses are populated in an exploration tree. A feature vector is maintained for each node in the graph. The ranking of topological hypotheses favours a topological structure with lesser vertices and a more completely explored section. The top N models are selected for further evolution. This approach is demonstrated for a graph of upto 30 nodes. According to results presented in this paper (Marinakis & Dudek 2010), for a graph of 50 nodes, the algorithm took approximately one hour to converge to a solution.

McClelland *et al* (2013)present the Qualitative Relationship Map (QRM) framework in which geometric relationships are encoded between the landmarks. The geometric relation between the landmarks are defined using a triplet of landmarks. This representation is called as Extended Double Cross (EDC). EDC is an extension of Freska's Double Cross (Freksa 1992) method. QRM assumes that landmarks are sufficiently distinct, either point-like or with known geometries. QRM also assumes that a relative ordering of distance of visible landmarks can be done. For any three landmarks A, B and C, EDC defines a qualitative state between C and a vector $\overrightarrow{AB}$, from point A to point B. C can be either on the right or the left of $\overrightarrow{AB}$, in front or behind of A or B, or in the direction of $\overrightarrow{AB}$. This qualitative representation allows one to capture angles in the triangle ABC. EDC also compares the distance of C to A against the distance from C to B, *etc.* This way, EDC adds explicit qualitative statements about edge lengths $|AB|$, $|BC|$, $|CA|$. By using unary and compositional operators on EDC, combinations of various states can be obtained. The map structure is represented as 3-uniform hypergraph. Each node corresponds to an observed landmark. Each edge connects three nodes. Each edge also contains a qualitative state which defines geometric features among these three nodes.

Another line of research is to learn from the spatial cognitive ability of living species and emulate it. RatSLAM (Milford & Wyeth 2008) is one such influential effort which tries to emulate the rodent's hippocampus. The hippocampus is an area in the brain of mammals that plays an important role in long-term memory and spatial navigation. This effort has established computational models inspired by the rodent hippocampus. The effort has demonstrated that such a SLAM solution works better than current state-of-the-art visual SLAM systems. A rodent maintains a representation of its own pose as certain cell types in its brain. An example of such cells is *place* cells, which get fired consistently when the agent is at a particular location in the environment, and not anywhere else. It has been demonstrated that Rats are also able to update and even "relocalize" their neural estimate of pose using external sensing such as vision, olfaction, and whisking. Unlike modern day robots, they do not build detailed geometrical representations of the environment. They rely on learnt associations between external perception and the pose belief created from integration of self-motion cues. The self-motion cues are rotation, forward speed and visual scene description. Other kind of neural cells discovered are *head direction* cells. These get fired when the rodent's head is at a certain global orientation. Another complex-behaviour exhibiting cell discovered is the *grid* cell. As explained in the paper on RatSLAM (Milford & Wyeth 2008), a single grid cell will fire when the rat is located at any of the vertices of a tessellating hexagonal pattern across the environment. There are also conjunctive grid cells – cells that fire only when the rat is at a certain location and facing in a specific orientation. All the cell types have two fundamental characteristics: (i) they are anchored to external landmarks and (ii) they persist in darkness. Experiments with landmark manipulation show that the rodent brain can use visual sighting of familiar landmarks to correct its pose estimation. A similar function is performed to the update process in robotic SLAM. Figure 14 represents broad connectivity between the various cells used in RatSLAM. As with robots, pose estimates degrade with time in the absence of external cues, suggesting similar computation must be part of the prediction process in robot SLAM. Results demonstrated by RatSLAM are very encouraging. It has demonstrated successful mapping over data-sets which involve a total path length of around 66 Km, with 44 loop closures. The largest loop closure had a size of around 5000m. An open source implementation of this algorithm is available at website www.openslam.org (Stachniss *et al* 2014).

FAB-MAP 3D (Paul & Newman 2010) is an evolution of FAB-MAP framework for appearance based navigation and topological mapping. It models a location as a collection of words (as

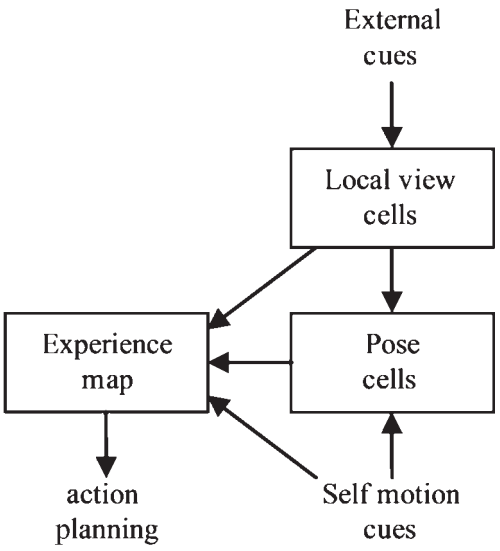

Figure 14. Broad Connectivity between RatSlam system (Milford & Wyeth 2008).

done in FAB-MAP) along with spatial information in terms of the range information for words. Hence, the algorithm has been able to cut down on false negative rate of loop closure detections of FAB-MAP.

5. Hybrid map generating approaches

Hybrid approaches combine the advantages of both metrical map generating solutions and qualitative methods. Most of the hybrid mapping approaches generate a global topological map of the environment and insert metrical data into the topological map. Each node in the topological map represents a distinct place or feature in the environment or a submap of the environment. Each edge represents topological relation or coordinate transformation between nodes. The loop closure is handled either at a metrical level or at topological level. As the robot explores the environment, a number of topological hypotheses satisfy current observations. These are usually arranged in a tree structure for further extensions. Since metrical maps are handled for smaller areas, complexity of the metrical map generating approaches remains manageable and within computational bounds.

In some approaches, each node in the topological map is a distinct place in the environment. The robot acts as the local observer during the exploration of the envorinment. It gathers observations continuously and at the same time avoids obstacles (Ko *et al* 2004; Tomatis *et al* 2002). In such approaches, the effort is focussed on finding qualitatively distinct locations from the metrical data. The problem with such an approach is that the number of locations increases as the exploration progresses. A number of detected distinct locations may refer to the same physical location. To maintain consistency, and in order to solve the loop closure problem, such locations should be suitably collapsed. Also, the count of detected distinct places should be kept as small as possible. This decision of when to add a new distinct place is difficult to characterize. Some methods use arbitrary heuristics; for example, reduce locations after every some meters of explorations (Zimmer 2000; McClelland *et al* 2013). Some approaches accept user inputs for location

definition (Thrun *et al* 1998). Other approaches try to model the full Bayesian distribution over a topological hypothesis (Blanco *et al* 2008a). They make use of topological constraints to create a consistent global topological map. The efforts focus is on solving the loop closing hypothesis by minimizing the possible number of places (qualitatively distinct locations) in the world. Each possible hypotheses is assigned a probability value. Only high probable candidate maps are evolved with further observations.

In some approaches, each node of the topological map represents a small metric map of the environment (Bosse *et al* 2003; Lisien *et al* 2005; Tully *et al* 2012; Beeson *et al* 2010). The edges between the nodes encodes the path traveled by the robot from the center point of one metric map to the center point of the next metric map. In the Atlas framework (Bosse *et al* 2003), each node in the topological graph represents a submap of fixed size. Each submap maintains a local coordinate frame. Each edge represents the transformation between the connecting frames with uncertainty. For transformation between arbitrary frames, Dijkstra's projection is used. A Dijkstra's projection is a global arrangement of the frames using composition along the Dijkstra's shortest path. Loop closures are handled using efficient matching based on the map's features matching. For selection of topological hypothesis among candidates, a performance metric is used. This metric is based on how much is the robot uncertainty and how well this hypothesis explains the sensor data.

In the hierarchical atlas (HSLAM) (Lisien *et al* 2005) approach, the topology is based on generalized voronoi graphs, known as Reduced GVG (RGVG) (Nagatani & Choset 1999). Each node contains degree and equidistant value. When robot travels along an edge, a feature based map is generated. The authors suggest any method can be used for this purpose. Mahalanobis distance is used to compare the node and the edge maps for narrowing down on possible topological configurations. The loop closures are handled at the topological level.

In Tully *et al* (2012), Tully *et al* present a unified filtering framework for hybrid SLAM and localization of robot in metric/topological maps. The hybrid map representation is based on the hierarchical atlas (Lisien *et al* 2005). For each topological hypothesis in the topological forest, an Extended Kalman filter is used to estimate the robot location and a metrical submap. Each metrical map is of fixed sizes and loop free. The loop closures are handled at topological levels. A hypothesis forest is maintained where each node in it represents the candidate topological hypotheses. For each robot action, topological hypotheses are extended based on the status of the current vertex. The best topological hypotheses is chosen based on the value of posterior probability of each hypothesis given a sequence of sensor measurements and robot motion inputs. The intuition is that the correct hypothesis will better fit the sensor measurements, therefore better equipped to estimate the true robot state. A penalty is imposed for wrong observation of landmarks locations. A high penalty is imposed to a topological hypothesis if it is not consistent with the environment observation *i.e.* the robot is in a different submap. A graph which explains the sensor data with the simplest representation (lesser number of nodes and edges) is preferred. This helps in avoiding over-fitting of data. The pruning of hypotheses is based on a number of tests which includes degree test, planarity test, likelihood update test and posterior probability test. A garbage collector hypothesis is used to represent all those hypotheses which have been pruned. If by mistake, a correct hypothesis is pruned, likelihood of garbage collector hypothesis will become high. This will signal the algorithm to backtrack as the correct hypothesis has been wrongly pruned.

The *hybrid spatial semantic hierarchy* (HSSH) (Beeson *et al* 2010; Kuipers *et al* 2004) is another important work based on SSH approach. It extends SSH (Kuipers *et al* 2000) and makes strong assumptions about the sensory capability of a navigational agent. It assumes the kind of sensors available, like range sensor, which can be used for distinct place recognition through

controlled movements of the robot. These assumptions allow the basic SSH to be extended by using various existing metrical mapping techniques to create a precise metrical model of the local surroundings (Beeson *et al* 2010). The HSSH framework is shown in figure 15. At the *Local Metrical Level* a small-space local map of the environment, known as Local Perceptual Map (LPM), is created. This can be done using the latest metrical map generating approaches. It allows the robot to be at a distinct place quickly. Using this LPM, the robot localizes itself and uses it for local navigation and hazard avoidance. In the next level, the *Local Topological Level*, the robot identifies discrete places in the environment and symbolically describes the paths through these places. This is called the local topological map. This topological map is extracted from the LPM using *gateways*. A gateway is a place that makes a particular region qualitatively distinct from another. For example, a door in the room is a gateway as it differentiates between the room region and the outside room region. At the *Global Topological Level*, the robot combines these local topological maps, resolves structural ambiguities and describes the environment in terms of path, places and regions in the best possible manner. The *Global Metrical Level* specifies a layout of the environment (places, paths, regions) in a single frame of reference. With the use of the LPM generated at distinct places and the topological skeleton structure, global metrical maps are evolved.

An approach which uses hybrid metric-topological maps for navigation is discussed in Marder-Eppstein *et al* (2011). It uses topological maps overlaid with local occupancy grids. Figure 3a is an example of this. The aim of this paper is to provide effective and fast navigation using hybrid map rather than maintaining an accurate hybrid map at every instant. It assumes a

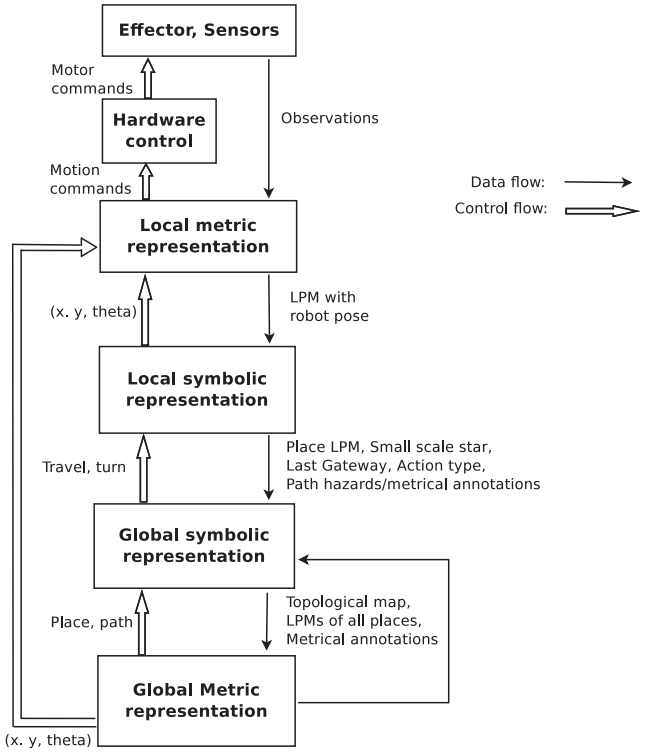

Figure 15. Hybrid-SSH framework (Beeson *et al* 2010).

hybrid map as the input. A plan is formed over the topological map and followed in local grids of correspondent nodes. As changes are observed in the map, occupancy grids are overlaid. To avoid long time-delays, only the grids in the immediate neighbourhood of the change are recomputed. Other grids are recomputed only on demand, for example, when an area is visited by the robot. For each change in position of the node or addition of a new node, occupancy grids are overlaid.

Some application scenarios, such as mines, disaster areas, may allow placement of RFID tags in the environment. These tags may be already existing in the environment or placed during the robotic operation. In Forster *et al* (2013), Christian Forster *et al* present a Radio Frequency Identification (RFID) tags-based hybrid mapping approach. The features of the environment are perceived using RFID tags. This approach maintains a weighted graph, called co-occurrence graph. This graph can be extended in an online fashion. Each node in this graph stands for observed RFID tags. Edge weight between two nodes reflects an estimate of how closely these two tags are located. This closeness is computed using radio signal strength (RSS). Using min-Ncut criterion (Malik & Shi 2000), a group of RFID-tags is grouped as one virtual topological node. Particle filter based mapping is used for building metric submap of RFID tag location. SLAM is performed only at a topological level. For localization purposes, an active topological node along with the metrical submap is always maintained in which the robot is currently located.

In Oberländer *et al* (2008), the authors propose a FastSLAM based method to generate topological and semantic maps of the environment. Environment is represented as using a new type of feature, named *regions*. A region is a feature descriptor which approximates the environment with a set of rectangles. Topological relations between the regions are stored using links between regions. Each region is associated with a semantic tag and a confidence value. Each link also has a confidence value associated with it.

In T-SLAM (Ferreira *et al* 2008), the authors present a method to integrate topological and metrical maps. The assumption made is that the topological and metric maps are created independently. The metrical maps are registered with topological map nodes.

6. SLAM in dynamic environments

All the above mentioned methods assume a static environment of operation, which is obviously not true for real world deployments. Two types of dynamic objects can be identified (i) Type I: High dynamic objects or entities which should not be part of the map *e.g.* cars and moving people. These objects should be identified and removed during the mapping process (*ii*) Type II: Low dynamic objects or entities which should be part of the map. Such objects move with low frequency or are so slow that their movement is invisible to the robot *e.g.* door (open or closed), furniture and items stored in a warehouse. Mapping process needs to maintain different configuration of such objects for capturing this dynamism. The different dynamic objects may move or change at different rates, gradually or adruptly. SLAM methods for static environments can be adapted to use in dynamic environments by providing appropriate treatment of the dynamic objects. The correct treatment of dynamic objects is crucial for long-term operation specially when the robot makes multiple passes over the same environment. Otherwise, the SLAM module could fail because of incorrect localization and wrong data association due to dynamic objects.

For SLAM in dynamic environments, Type I dynamic objects are continuously detected and tracked (Hähnel *et al* 2003; Wang *et al* 2002; Montemerlo & Thrun 2002; Zhao *et al* 2008). The dynamic objects could be moving people, moving cars, *etc.* The environmental sensor measurements are processed to filter out the dynamic objects. The resultant measurements are used

for the SLAM process. In Huang & Wong (2005), authors have presented a unified solution for online SLAM and multi-target tracking. This approach based on filtering out of dynamic objects from sensor measurements is effective only for highly dynamic environments.

Some approaches maintain multiple spatial maps for the dynamic environment. Denis F Wolf *et al* (Wolf & Sukhatme 2003; Wolf & Sukhatme 2004) maintain two different occupancy grids for the static and the dynamic part of the environment, given the robot's true location. The union of these two maps provides a complete information about the environment. In Wolf & Sukhatme (2004), a third map, representing static landmarks in the environment, is also maintained. This map is used for correct robot localization. In Biber & Duckett (2009), Peter *et al* model the environment using multiple time-scale maps to capture the dynamism of the environment. Each map is represented as a set of samples taken from the measurements. The set of different time-scale maps allows the representation of different hypotheses related to dynamic objects. Such modelling of the environment is more suited to handle Type II dynamic objects. This method makes the system robust to outliers. Each time-scale map is updated with a different time-scale specific learning rate. The intuition is that old memories should fade at different rates for different time scales (Biber & Duckett 2009). For localization, the robot compares current observation with all the maps and chooses the map which better fits the current observations.

Some dynamic objects may not be moving at the time of measurements *e.g.* parked cars or objects such as people and cars at traffic intersections which may be still for some time *e.g.* as explained in Zhao *et al* (2008). Some objects may move with a relatively low frequency. The rejection of all dynamic objects will not be helpful in such cases, as such low-frequency moving objects can help in localization. In Biswas *et al* (2002), authors assume objects move sufficiently slowly that they can be assumed to be static during the mapping process. In this paper, an occupancy grid mapping algorithm called Robot Object Mapping Algorithm (ROMA) is explained. ROMA is based on the Expectation-Maximization (EM) algorithm, ROMA builds a static occupancy grid map of the environment at different operator specified times. The changes in the environment are identified by taking the map differences. Zhao *et al* (2008) take a different approach for change detection. In this approach, the laser scan data is divided into clusters. Each cluster from sensor observations is identified as static, moving, unknown class (called seed) and newly detected objects. The classification of seed class is done at the end of each iteration using its size, shape and history of motion. If the seed remains static during the operations, only then it is added to the map.

Daniel *et al* (Meyer-Delius et al 2010) propose the use of temporary local maps to enable robust localization. The temporary maps represents the dynamic entities of the environment. The objects which change their location with low frequency are labelled as semi-static objects (Meyer-Delius *et al* 2010) or low-dynamic objects (Walcott-Bryant *et al* 2012). The example of semi-static objects is parked cars in parking lots and stored items in a warehouse (Meyer-Delius *et al* 2010). The temporary local maps represent the measurements due to semi-static objects in the environment. These maps are created using HOG-Man (Grisetti *et al* 2010b). In this paper, it is assumed that Type I dynamic objects are detected and filtered out. It is also assumed that the static map of the environment is available. For the purpose of localization, the temporary maps augment the static map of the environment for localization.

Cyrill et al (Stachniss & Bougard 2005) present a different approach to represent dynamic environment by explicitly modelling low-dynamic objects' (Type II) states. This allows capture of various possible configuration of the environment. For example, a door can be either open or close. The possible states or configurations of dynamic objects becomes part of the map. The environment is divided into local sub-maps with the assumption that dynamic aspects of one

sub-map is independent of neighbouring sub-maps. The dynamic aspects are modelled in each sub-map.

Dynamic Pose Graph SLAM (DPG-SLAM) (Walcott-Bryant *et al* 2012) incorporates time dimension into the mapping process. The movement of Type II dynamic objects as observed during different passes of the same environment is modelled. DPG-SLAM maintains two different maps (i) *active map*: a current representation of the static part of the environment and (ii) *dynamic map*: a map of dynamic objects. This algorithm also maintains a connected graph, which represents the pose chain sequence of all previous passes. The graph is called as Dynamic Pose Graph (DPG). The active and dynamic maps are continuously updated for observed changes at each location from the previous pass. iSAM (Kaess *et al* 2008) is used as a state estimation engine. As robot makes more passes of the environment, the size of the DPG grows. This approach keeps the size of the DPG tractable by removing the inactive nodes in the graph. These inactive nodes refer to removed data from the current active map *i.e.* observation corresponding to previous locations of the dynamic objects.

For a known fixed environment, sensors in the environment can also be used to detect dynamic objects (Cortés 2009; Zou & Tan 2013). These sensors complement the on-board sensing of the robot. CoSLAM (Zou & Tan 2013) is a collaborative slam where multiple cameras are present in the environment. These cameras may be static or moving independently. Feature points are selected from cameras' images and tracked using the Kanade-Lucas-Tomasi (KLT) (Shi & Tomasi 1994; Tomasi & Kannade 1991) tracker. The dynamic points are detected as one with large re-projection errors between $(n-1)^{\text{th}}$ to n^{th} frame. A dynamic point is considered to become static if its projection has below a threshold re-projection errors for a predefined number of continuous frames. This processed information from network sensors is communicated to the robot for various usage.

In Ramdev *et al* (2013) & Kundu *et al* (2011), authors have discussed an incremental visual SLAM (VSLAM) solution using a monocular camera. From the input image sequence, high dynamic objects are segmented. A separate VSLAM module processes each moving object. A Bearing-Only-Tracking (BOT) (Kundu *et al* 2011) algorithm is used to track the moving targets. This module provides the movement trajectory of the moving object. One VSLAM module processes the static contents from the image sequence. The output of this module is reconstruction of the static world and trajectory of the camera movement. In this map, no dynamic object is present. To incorporate the trajectory of the moving objects, these outputs are combined together after finding relative scale of the object *w.r.t*. the stationary world. The only reported degenerate case of this approach is when the movement of camera and objects is correlated *e.g.* both are moving parallel in a straight line motion.

7. DataSets

For SLAM experimentation, good datasets capture is an important activity. Table 3 shows various open published datasets. In this table, The multiplicity of any sensor is captured using 'x' character. For example, 2xO represents that two omni directional cameras are present. IR, S, M, O, B/W stands for infra-red, stereo, monocular, omni-directional and black-and-white monocular cameras respectively. 3D and 2D stands for three dimentional and two dimentional range scanner respectively. Y and N stands for Yes and No respectively. D-GPS stands for differential GPS.

Table 1. SLAM algorithms for static environments.

Algorithm type	Algorithms' name/references
Metrical map generating approaches	EKF-SLAM (Guivant & Nebot 2001; Thrun *et al* 2006), UKF (Thrun *et al* 2006), FastSLAM (Montemerlo *et al* 2002; Montremerlo *et al* 2003), (Fox 2003), DP-SLAM (Eliazar & and Parr 2003; Eliazar & Parr 2004), D-SLAM (Wang *et al* 2007a), SEIF (Thrun *et al* 2004; Eustice *et al* 2005), ESEIF (Walter *et al* 2005), Mono-SLAM (Davison *et al* 2007), tinySLAM (Steux & Hamzaoui 2010), Divide & Conqueor SLAM (Paz *et al* 2008), FAB-MAP (Cummins & Newman 2010), OctoMap (Wurm *et al* 2010), (Kerl *et al* 2013), (Endres *et al* 2012) $\sqrt{SAM}$ (Frank Dellaert & Michael Kaess 2006), iSAM (Kaess *et al* 2008), iSAM2 (Kaess *et al* 2012), RISE (Rosen *et al* 2012), (Rosen *et al* 2013), (Thrun & Montemerlo 2006), HOG-Man (Grisetti *et al* 2010b), SGD (Olson *et al* 2006), (Grisetti *et al* 2009), TJTF (Paskin 2003), g^2o (Kümmerle *et al* 2011), Max-Mixtures (Olson & Agarwal 2013), RRR (Latif *et al* 2012a), iRRR (Latif *et al* 2012b) Switch Constraints (Sünderhauf & Protzel 2012), MoG (Pfingsthorn & Birk 2012), SPA (Konolige *et al* 2010), T-SAM (Ni *et al* 2007), Treemap (Frese 2006), SLSJF (Huang *et al* 2008a), Linear SLAM (Shao *et al* 2013), (Andrew Howard & Matarić 2006), (Saeedi *et al* 2011), (Rizzini & Caselli 2010), D-SLAM (Wang *et al* 2007a), C-SAM (Andersson & Nygards 2008), DDF-SAM (Cunningham *et al* 2010; Cunningham *et al* 2013), (Reid & Braünl 2011), (Olson *et al* 2013), COP-SLAM (Dubbelman & Browning 2013), (Kretzschmar & Stachniss 2012), Loopy-SLAM (Ranganathan *et al* 2007)
Topological map generating approaches	SSH (Kuipers *et al* 2000), RAT-SLAM (Milford & Wyeth 2008), PTM (Ranganathan *et al* 2006), GVG (Choset & Burdick 1995), SGVG (Tao *et al* 2011), (Marinakis & Dudek 2010), QRM (McClelland *et al* 2013) FAB-MAP 3D (Paul & Newman 2010)
Hybrid map generating approaches	HSSH (Beeson *et al* 2010), (Marder-Eppstein *et al* 2011), Hierarchical SLAM (Estrada *et al* 2005), Atlas (Bosse *et al* 2003), HSLAM (Lisien *et al* 2005), (Tully *et al* 2012), HMT-SLAM (Blanco *et al* 2008a), (Oberländer *et al* 2008), T-SLAM (Ferreira *et al* 2008), (Forster *et al* 2013)

Table 2. SLAM algorithms for dynamic environments.

Dynamic object types	Algorithms' Names/References
Type I: High dynamic objects	(Hähnel *et al* 2003; Wang *et al* 2002; Montemerlo & Thrun 2002) (Zhao *et al* 2008; Huang & Wong 2005) (Zou & Tan 2013; Ramdev *et al* 2013; Kundu *et al* 2011)
Type II: Low dynamic objects	(Wolf & Sukhatme 2003; 2004), ROMA (Biswas *et al* 2002), (Stachniss & Bougard 2005), DPG-SLAM (Walcott-Bryant *et al* 2012), CoSLAM (Zou & Tan 2013), (Ramdev *et al* 2013; Kundu *et al* 2011)

Table 3. Details of open dataset. Refer 7 for legends.

Sr. no.	Dataset	Range sensor	Vision	RGB-D	Odom	IMU	GPS	Misc. remarks
1	Barcelona Lab Dataset (Catalunya 2012)	3D	S	N	Y	Y	N	Camera network
2	The Gravel Pit Lidar-Intensity Imagery Dataset (Anderson *et al* 2012)	Nod,2D	N	N	N	N	D	–
3	The Denvor Island Rover Navigation Dataset (Furgale *et al* 2012)	3D	S	N	N		D incliometer present.	Sun sensor,
4	SLAM Benchmarking Dataset (Burgard *et al* 2009; Kümmerle et al2009)	x2D	N	N	Y	Y	N available	Pose graphs
5	Malága 2009 (Blanco *et al* 2009)	2x2D SICK, 2x2D Hokoyu	Y	N	N	3x	Y	–
6	PeRL (Pandey *et al* 2011)	3D, 2x2D	O	N	Y	2x	Y GPS-IMU	Integrated
7	Malàga Urban Dataset (Blanco *et al* 2014)	2x2D SICK, 3x2D Hokoyu	S	N	N	Y	Y	–
8	RAWSEEDS (RAWSEEDS 2009)	2x2D	S,O, 2xB/W	N	Y	Y	Y	–
9	KITTI (Geiger *et al* 2013; Fritsch *et al* 2013; Geiger *et al* 2012)	3D	S, B/W	N	N	N	Y	HD-cameras
10	The Canadian Planetary Emulation Terrain 3D Mapping Dataset (Tong *et al* 2013)	2D	S	N	Y	Y	D-GPS	*
11	Victoria Park (Nebot 2000)	2D	O	N	Y	N	Y	–
12	Radish Dataset Repository (Howard & Roy 2003)	2D	N	N	Y	N	N indoor buildings	Mostly
13	Robotic 3D Scan Repository (Borrmann & Nüchter 2014)	3D	Y	N	Y	Y	Y	*, IR camera
14	The UTIAS Multi-robot Cooperative Localization and Mapping Dataset (Leung *et al* 2011)	N	M	N	Y	N	N	Ground truth using Vicon motion capture system
15	The Marulan Data Sets (Peynot *et al* 2010)	4x2D	S, IR	N	Y	Y	Y wave radar	includes milliMeter
16	COLD The CoSy Localization Database (Pronobis & Caputo 2009)	2D	M, O	N	Y	N	N	–

Table 3. *contd.*

Sr. no.	Dataset	Range sensor	Vision	RGB-D	Odom	IMU	GPS	Misc. remarks
18	The New College Vision and Laser Data Set (Smith *et al* 2009)	2x2D	S,O	N	Y	N	Y	–
19	MIT DARPA Urban Challenge Datase (Huang *et al* 2010a)	3D, 12x2D	5xM	N	Y	Y	Y	–
20	Navlab SLAMMOT Datasets (Wang *et al* 2004)	2D	M	N	Y	N	N	–
23	Annotated Laser Data Set (Yang *et al* 2011)	2D	Tri- Camera	N	Y	N	N output of tri-camera	A single Image
24	RGB-D Dataset (Nathan Silberman *et al* 2012)	N	N	Y	N	N	N	–
25	ICL-NUIM (Handa *et al* 2014)	N	N	Y	N	N	N	–

8. Conclusion

This paper presents a survey on SLAM algorithms for unmanned ground robots using the literature presented till December, 2013. The characteristics of algorithms for static and dynamic environments are presented. The classification of algorithms is based on metrical, topological and hybrid map generating approaches and these are presented for static environments. Metrical map generating algorithms for multi-robot SLAM are also discussed. Table 1 summarizes these approaches. The Bayesian Filter based approaches are useful for structured and small environments. The graph-based SLAM approaches are useful for larger environments and for robots with sufficient compute capabilities. The current research on SLAM focusses on evolving graph-based SLAM algorithms for robust and online operations in large ambiguous environments. Metrical map generating approaches do not scale well. For metrical maps, we presented bayesian filter based and graph-based approaches. This paper also presents the treatment of dynamism in the environment by various methods. The current research on SLAM also focusses evolving algorithms for long-term autonomous operation of the robot in dynamic environments. Table 2 summarizes the algorithms for dynamic environments. Table 3 provides a detailed list of open and published datasets. Semantic maps provide labels to map objects. This labelling can be useful in high-level task planning and communication with the robot. This paper touches upon some simple semantically enabled approaches. A more comprehensive discussion on semantic maps will be attempted as a future work.

References

Agarwal P and Olson E 2012 Variable reordering strategies for slam. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3844–3850, Vilamoura

Aggarwal P, Tipaldi G D, Spinello L, Stachniss C and Bougrad W 2013 Robust map optimization using dynamic covariance scaling. In: *Proceedings Of 2013 IEEE International Conference on Robotics and Automation (ICRA)*, Karlsruhe, Germany

Anderson A, McManus C, Dong H, Beerepoot E and Barfoot T D 2012 *The gravel pit lidar-intensity imagery dataset.* Technical Report ASRL-2012-ABL001, University of Toronto

Andersson L A A and Nygards J 2008 C-SAM: Multi-robot slam using square root information smoothing. In: *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2798–2805, Pasadena, CA, USA

Andrew Howard G S S and Matarić M J 2006 Multi-robot mapping using manifold representations. *Proc. IEEE – Special Issue Multi-robot Syst.* 94(9): 1360–1369

Bailey T, Dissanayake G and Durrant-Whyte H 2000 A computationally efficient solution to the simultaneous localisation and map building (SLAM) problem. In: *Proceedings of IEEE International Conference on Robotics and Automation*, vol. 2, pages 1009–1014

Bailey T and Durrant-Whyte H 2006 Simultaneous localisation and mapping (SLAM): Part II. *IEEE Robot. Autom. Mag.* 13(3): 108–117

Bailey T, Nieto J, Guivant J, Stevens M and Nebot E 2006 Consistency of EKF-SLAM algorithm. In: *Proceedings of IEEE/RSJ Conference on Intelligent Robotics Systems*, pages 3352–3358

Barrera T, Hast A and Bengtsson E 2004 Incremental spherical linear interpolation. In: *Proceedings SIGRAD*, vol. 13, pages 7–13

Bay H, Ess A, Tuytelaars T and Gool L V 2008 Speeded-up robust features SURF. *Comput. Vis. Image Understanding* 110(3): 346–359

Beeson P, Modayil J and Kuipers B 2010 Factoring the mapping problem: Mobile robot map-building in the hybrid spatial semantic hierarchy. *Int. J. Robot. Res.* 29(4): 428–459

Biber P and Duckett T 2009 Experimental analysis of sample-based maps for Long-Term SLAM. *Int. J. Robot. Res.* 28(1): 20–33

Biswas R, Limketkai B, Sanner S and Thrun S 2002 Towards object mapping in non-stationary environments with mobile robots. In: *Proceedings of 2002 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1014–1019, EPFL, Lausanne, Switzerland

Blanco J, Ferandez-Madrigal J and Gonzalez J 2008a Toward a unified bayesian approach to hybrid metric-topological SLAM. In: *IEEE Transactions on Robotics*, vol. 24, pages 259–270

Blanco J-L 2014 Mobile robotics programming toolkit. URL www.mrpt.org

Blanco J-L, Fernández-Madrigal J-A and Gonzàlez J 2008b Efficient probabilistic range-only SLAM. In: *Proceedings of 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1017–1022, Nice, France

Blanco J-L, Moreno F-A and González J 2009 A collection of outdoor robotic datasets with centimeter-accuracy ground truth. *Autonom. Robots* 27(4): 327–351

Blanco J-L, Moreno F-A and González-Jiménez J 2014 The málaga urban dataset: High-rate stereo and lidars in a realistic urban scenario. *Int. J. Robot. Res.* 33(2) http://www.mrpt.org/MalagaUrbanDataset accessed on 2-SeP-2014]

Borrmann D and Nüchter A 2014 *Robotics 3D scan repository.* http://kos.informatik.uni-osnabrueck.de/3Dscans/ online accessed on 04-September 2014

Bosse M, Newman P, Leonard J and Soika M 2003 An atlas framework for scalable mapping. In: *Proceedings of IEEE International Conference on Robotics and Automation*, vol. 2, pages 1899–1906, Taipei, Taiwan

Burgard W, Stachniss C, Grisetti G, Steder B, Kuemmerle R, Dornhege C, Ruhnke M, Kleiner A and Tardos J D 2009 A comparison of slam algorithms based on a graph of relations. In: *Proceedings of IEEE/RSJ Conference on Robots and Systems (IROS)*

Cadena C, Gàlvez-Lòpez D, Tardòs J D and Neira J 2012 Robust place recognition with stereo sequences. *IEEE Trans. Robot.* 28(4): x–y

Castellanos J A, Martinez-Cantin R, Castellanos J A and Neira J 2007 Robocentric map joining: Improving the Consistency of EKF-SLAM. In: *Robotics and Autonomous Systems* vol. 55, pages 21–29

Castellanos J A and Neira J 2004 Limits to consistency of EKF-SLAM algorithm. In: *5th IFAC Symposium on Intelligent Autonomous Vehicles*, pages 3562–3568

Catalunya U P D 2012 *Barcelona robot lab dataset.* URL http://www.iri.upc.edu/research/webprojects/pau/datasets/BRL/ *online accessed on 02-September-2014*

Choset H and Burdick J 1995 Sensor based planning, Part I: The Generalized Voronoi Graph. In: *Proceedings of the 1995 IEEE International Conference on Robotics and Automation (ICRA '95)*, vol. 2, pages 1649–1655

Chow C and Liu C 1968 Approximating discrete probability distribution with the dependencies trees. *IEEE Trans. Inf. Theory* 14(3): 462–467

Clemente L, Davison A, Ried I and Neira J 2007 Mapping Large Loops with a Single Hand-Held Camera. In: *Proceedings of Robotics: Science and Systems III*. URL http://www.roboticsproceedings.org/rss03/index.html

Cortés A S 2009 URUS project: Communication systems. In: *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, Workshop on Network Robot Systesm*, pages 1–44

Cowell R, David A, Lauritzen S and Spiegelhalter D 1999 *Probabilistic Networks and Expert Systems.* Springer-Verlag

Cummins M and Newman P 2008 FAB-MAP: Probabilistic localization and mapping in the space of appearance. *Int. J. Robot. Res.* 27(6): 647–665

Cummins M and Newman P 2009 Highly scalable appearance-only SLAM – FAB-MAP 2.0. In: *Robot. Sci. Syst. (RSS)*, Seattle, USA

Cummins M and Newman P 2010 Appearance-only SLAM at Large Scale with FAB-MAP 2.0. *Int. J. Robot. Res.* 30(9): 1100–1123

Cunningham A, Indelman V and Dellaert F 2013 DDF-SAM 2.0: Consistent distributed smoothing and mapping. In: *IEEE Intl. Conference on Robotics and Automation (ICRA)*, pages 5220–5227, Karlsruhe; Germany

Cunningham A, Paluri M and Dellaert F 2010 DDF-SAM: Fully distributed SLAM using constrained factor graphs. In: *IEEE International Conference on Intelligent Robotics and Systems (IROS)*, pages 3025–3030

Cunningham A, Wurm K, Burgard W and Dellaert F 2012 Fully distributed scalable smoothing and mapping with robust multi-robot data association. In: *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*

Davis T, Gilbert J, Larimore S and Ng E 2004 A column approximate minimum degree ordering heuristic. *ACM Transactions on Math Software* 30(3): 353–376

Davison A *Homepage.* http://www.doc.ic.ac.uk/ajd/. [online accessed on 04-September-2014]

Davison A J, Reid I D, Molton N D and Stasse O 2007 MonoSLAM: Real-time single camera SLAM. *IEEE Trans. Pattern Anal. Mach. Intell. (PAMI)* 26(6): 1052–1067

Deans M and Hebert M 2000 Experimental comparision of techniques for localization and mapping using bearing only sensor. In: *Proceedings of IEEE Symposium on Experimental Robotics*, vol. 271, pages 395–404

Deans S R 1983 *The radon transforms and some of its applications*. New York: John Wiley and Sons

Dhiman N K, Deodhare D and Khemani D 2012 A review of mapping technologies for autonomous mobile robot systems. In: *Proceedings of the 5th ACM COMPUTE Conference: Intelligent & scalable system technologies*

Doucet A, de Freitas J, Murphy K and Russel S 2000 Rao blackwellised particle filtering for dynamic bayesian networks. In: *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 176–183, Stanford, CA, USA

Doucet A, de Freitas N and Gordon N 2001 *An introduction to Sequential Monte Carlo Methods in Practice*. Springer

Dubbelman G and Browning B 2013 Closed-form online pose-chain slam. In: *Proceedings of IEEE International Conference on Robotics and Automation*, pages 5190–5197, Karlsruhe, Germany

Dubbelman G, Dorst L and Pijls H 2010 Efficient trajectory bending with applications to loop closures. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4836–4842, Taipei, Taiwan

Durrant-Whyte H and Bailey T 2006 Simultaneous Localisation and Mapping (SLAM): Part I The Essential Algorithms. *IEEE Robot. Autom. Mag.* 13(2): 99–110

Eliazar A and and Parr R 2003 DP-SLAM: Fast, Robust Simultaneous Localization and Mapping without Predetermined Landmarks. In: *Proceedings 18th Int. Joint Conf. on Artificial Intelligence (IJCAI-03)*, pages 1135–1142. Morgan Kaufmann

Eliazar A and Parr R 2004 DP-SLAM 2.0. In: *IEEE International Conference on Robotics and Automation*, vol. 2, pages 1314–1320

Endres F, Hëss J and Engelhard N 2012 An evaluation of the rgb-d slam system. In: *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1691–696, Saint Paul, MN

Estrada C, Neira J and Tardos J D 2005 Hierarchical SLAM: Real-time accurate mapping of large environments. In: *IEEE Transactions on Robotics*, vol 21(4). pages 588–596

Eustice R, Walther M and Leonard J 2005 Sparse extended information filters: insights into sparsification. In: *Proceedings of the IEEE International Conference on Intelligent Robotics and Systems (IROS)*, pages 3281–3288

Ferreira F, Amorim I, Rocha R and Dias J 2008 T-slam: Registering topological and geometric maps for robot localization in large environments. In: *Proceedings of IEEE Conference on Multisensor Fusion and Integration for Intelligent Systems*, pages 392–398

Forster C, Sabatta D, Siegwart R and Scaramuzza D 2013 RFID-based hybrid metric-topological slam for GPS-denied environments. In: *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, pages 5228–5234, Karlsruhe, Germany

Fox D 2003 Adapting the sample size in particle filters through kld-sampling. *Int. J. Robot. Res.* 22: 985–1003

Frank Dellaert and Michael Kaess 2006 Square Root SAM: Simultaneous Location and Mapping via Square Root Information Smoothing. *Int. J. Robot. Res. (IJRR)* 25(12): 1181 Special issue on RSS 2006

Freksa C 1992 Using orientation information for qualitative spatial reasoning. In: *Theories and Methods of Spatio-Temporal Reasoning in Geographic Space (LNCS)*, vol. 639, pages 162–178

Frese U 2006 Treemap: An o(log n) algorithm for indoor simultaneous localization and mapping. *J. Autonom. Robot.* 21(2): 103–122

Frese U and Schröder L 2006 Closing a million landmark loop. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5032–5039, Beijing

Fritsch J, Kuehnl T and Geiger A 2013 A new performance measure and evaluation benchmark for road detection algorithms. In: *International Conference on Intelligent Transportation Systems (ITSC)*

Fuentes-Pacheoco J, Ruiz-Ascencio J and Rendón-Mancha J M 2012 Visual simultaneous localization and mapping: a survey. *Artif. Intell. Rev.* 43(1): 55–81

Furgale P T, Carle P J F, Enright J and Barfoot T D 2012. *Int. J. Robot. Res.* 31(6): 707–713 URL http://asrl.utias.utoronto.ca/datasets/devon-island-rover-navigation/ online accessed on 03-September-2014]

Galvez-Lopez D and Tardos J D 2012 Bags of binary words for fast place recognition in image sequences. *IEEE Trans. Robot.* 28(5): 1188–1197

Geiger A, Lenz P, Stiller C and Urtasun R 2011 Stereoscan: Dense 3D reconstruction in real time. In: *Intelligent Vehicle Symposium*

Geiger A, Lenz P, Stiller C and Urtasun R 2013 Vision meets robotics: the KITTI dataset. *International Journal of Robotics Research (IJRR)*

Geiger A, Lenz P, Stiller C and Urtasun R 2014 *KITTI vision benchmark dataset, odometry evaluation webpage.* [URL https://cvlibs.net/datasets/kitti/eval_odometry.php accessed on 2-September-2014]

Geiger A, Lenz P and Urtasun R 2012 Are we ready for autonomous driving? the KITTI vision benchmark suite. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*

Georgio F D *et al* Georgia Tech Smoothing And Mapping (GTSAM) download page. URL https://collab.cc.gatech.edu/borg/gtsam/. [online accessed on 03-September-2014]

Golub G and Loan C V 1996 *Matrix computations*. Baltimore: Hopkins University Press

Grisetti G, Kümmerle R, Stachniss C and Bougard W 2010a A tutorial on graph-based slam. *IEEE Intelligent Transportation System Magazine* 2(4): 31–43

Grisetti G, Kuṁmerle R, Stachniss C, Frese U and Hertzberg C 2010b Hierarchical Optimization on Manifolds for Online 2D and 3D Mapping. In: *Proceedings of the IEEE Intl. Conf. on Robotics and Automation*, pages 273–278, Anchorage, Alaska

Grisetti G, Stachniss C and Bougard W 2007 Improved Techniques for Grid Mapping with Rao-Blackwellized Particle Filters. *IEEE Trans. Robot.* 23(1): 34–46

Grisetti G, Stachniss C and Burgard W 2009 Nonlinear constraint network optimization for efficient map learning. *IEEE Trans. Intell. Transport. Syst.* 10(3): 428–439

Guivant J E and Nebot E 2001 Optimization of simultaneous localization and map-building algorithm for real time implementations. In: *IEEE Trans. Robot. Autom.*, vol. 17, pages 242–257

Hähnel D, Triebel R, Burgard W and Thrun S 2003 Map building with mobile robots in dynamic environments. In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*

Handa A, Whelan T, McDonald J and Davison A 2014 A benchmark for RGB-D visual odometry, 3D reconstruction and SLAM. In: *IEEE Intl. Conf. on Robotics and Automation, ICRA*, Hong Kong, China. (to appear)

Hendrickson B 1995 A multilevel algorithm for partitioning graphs. In: *Proceedings of ACM International Conference on Supercomputing*, pages 626–657, Sorrento

Hornung A, Wurm K M, Bennewitz M, Stachniss C and Burgard W 2013 OctoMap: An efficient probabilistic 3d mapping framework based on octrees. *Autonomous Robots*, pages 189–206. Software available at http://octomap.sf.net/

Howard A 2006 Multi-robot simultaneous localization and mapping using particle filters. *Int. J. Robot. Res.* 25(12): 1243–1256

Howard A and Roy N 2003 The robotics data set repository (Radish), [http://radish.sourceforge.net accessed on 04-September-2014]

Hu G, Huang S and Dissanayake G 2009 3D I-SLSJF: A consistent sparse local submap joining algorithm for building large-scale 3d maps. In: *Proceedings. of* 48^{th} *IEEE Conf. on Design and Control*, pages 6040–6045

Huang A, Antone M, Olson E, Fletcher L, Moore D, Teller S and Leonard J 2010a A high-rate, heterogeneous data set from the darpa urban challenge. *Int. J. Robot. Res.*, 29(13):1595–1601. [http://grandchallenge.mit.edu/wiki/index.php?title=PublicData online accesed on 04-September-2014]

Huang G Q and Wong Y K 2005 Online slam in dynamic environments. In: *Proceedings of 12th International Conference on Advanced Robotics (ICAR '05)*, pages 262–267, Seatle, WA

Huang S, Lai Y, Frese U and Dissanayake G 2010b How far is slam from a linear least squares problem? In: *Proceedings of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, pages 3011–3016, Taipei, Taiwan

Huang S, Wang Z and Dissanayake G 2008a Sparse local submap joining filter for building large-scale maps. *IEEE Trans. Robot.* 24(5): 1121–1130

Huang S, Wang Z, Dissanayake G and Frese U 2008b Iterated SLSJF: A sparse local submap joining algorithm with improved consistency. In: *Proceedings of 2008 Australiasan Conference on Robotics and Automation*, Canberra, Australia

Huang Joseph, David Millman M Q D S S T and Aggarwal A 2011 Efficient, Generalized Indoor WiFi GraphSLAM. In: *Proceedings of the IEEE Conference on Robotics and Automation*, pages 1038–1043, Shanghai, China

Kaess M 2008 *Incremental Smoothing And Mapping*, PhD thesis, Georgia Institute of Technology

Kaess M 2014 iSAM download webpage. [http://people.csail.mit.edu/kaess/isam accessed on 2-September-2014]

Kaess M, Ila V, Roberts R and Dellart F 2010 The bayes trees: An algorithmic foundation for probabilistic robot mapping. In: *Intl. Workshop on the Algorithmic Foundations of Robotics*, pages 157–173, Singapore

Kaess M, Johannsson H, Roberts R, Ila V, Leonard J and Dellaert F 2012 iSAM2: Incremental smoothing and mapping using the Bayes tree. *Int. J. Robot. Res.* 31: 216–235

Kaess M, Ranganathan A and Dellaert F 2008 iSAM: Incremental smoothing and mapping. *IEEE Trans. Robot.* 24(6): 1365–1378

Kalman R E 1960 A new approach to linear filtering and prediction problems. *Trans. ASME, Journal of Basic Engineering* 82: 35–45

Kerl C, Sturm J and Cremers D Compute Vision Group, TUM. [http://vision.in.tum.ne/data/datasets accessed on 04-September-2015]

Kerl C, Sturm J and Cremers D 2013 Dense visual slam for rgb-d cameras. In: *Proceedings of the Int. Conf. on Intelligent Robot Systems (IROS)*, pages 2100–2106, Tokyo

Kleiner A, Dornhege C and Dali S 2007 Mapping disaster area jointly: RFID-cordinated slam by humans and robots. In: *Proceedings of IEEE International Workshop on Safety, Security and Rescue Robotics*, pages 1–6

Kümmerle R, Steder B, Dornhege C, Ruhnke M, Grisetti G, Stachniss C and Kleiner A 2009 On measuring the accuracy of slam algorithms. *Journal of Autonomous Robots* 27(4): 387–407

Ko B Y, Song J B and Lee S 2004 Real-time building of a thinning-based topological map with metric features. In: *Proceedings of IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, pages 797–802, Japan

Kon 2009 Towards lifelong visual maps. In: *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1156–1163, address

Konolige K, Grisetti G, Kummerle R, Limketkai B and Vincent R 2010 Efficient Sparse Pose Adjustment for 2D Mapping. In: *Proceedings of the IEEE/RSJ Inl. Conf. on Intelligent Robot and Systems*, pages 22–29, Taipei

Kretzschmar H and Stachniss C 2012 Information theoretic compression of pose graph for laser based SLAM. *Int. J. Robot. Res. (IJRR)* 31(11): 1219–1230

Kschischang F R, Frey B J and Loeliger H A 2001 Factor graphs and the sum-product algorithm. *IEEE Trans. Inf. Theory* 29(10): 498–519

Kuemmerle R 2014 g^2o github repository. [http://github.com/RainerKuemmerle/g2o accessed on 2-September-2014]

Kuipers B, Browning R, Gribble B, Hewett M and Remolina E 2000 The spatial semantic hierarchy. *Artif. Intell.* 119: 191–233

Kuipers B, Modayil J, Beeson P, Macmohan M and Savelli F 2004 Learning metrical and global topological maps in the hybrid spatial semantic hierarchy. In: *Proceedings of IEEE International Conference on Robotics and Automation(ICRA)*, pages 4845–4851, Louisiana

Kümmerle R, Grisetti G, Strasdt H, Konolige K and Bougard W 2011 g^2o: A general framework for graph optimization. In: *Proceedings of the IEEE Conf. on Robotics and Automation*, pages 3607–3613, Shanghai

Kundu A, Krishana K M and Jawahar C V 2011 Realtime multibody visual slam with smoothly moving monocular camera. In: *Proceedings of IEEE International Conference on Computer Vision*, pages 2080–2086, Barcelona

Latif Y 2014 Github repository. [http://github.com/ylatif accessed on 2-September-2014]

Latif Y, Cadena C and 'e Niera J 2012a Robust loop closing over time. In: *Proceedings of Robotics: Science and Systems(RSS)*, Sydney, Australia

Latif Y, Cadena C and Neira J 2012b Realizing, reversing, recovering: Incremental robust loop closing over time using iRRR algorithm. In: *Proceedings of IEEE International Conference on Intelligent Robotics and Systems*, pages 4211–4217, Vilamoura

Latif Y, Cadena C and Neira J 2013 Robust loop closing over time for pose graph SLAM. *Int. J. Robot. Res.* 32(14): 1611–1626

Lee H-C, Lee S-H, Lee T-S, Kim D-J and Lee B-H 2012 A Survey of Map Merging Techniques for Cooperative- SLAM. In: *Proceedings of* 9^{th} *International Conference on Ubiquitous Robots and Ambient Intelligence(URAI)*, Daejeon, Korea

Lee J M 2003 *Introduction to smooth manifolds*, volume volume 218 of Graduate Text in Mathematics. Springer-Verlag

Leung K Y K, Halpern Y, Barfoot T D and Liu H H T 2011 The utias multi-robot cooperative localization and mapping dataset. *Int. J. Robot. Res.* 30(8): 969–974

Lisien B, Morales D, Silver D, Kantor G, Rekleitis I and Choset H 2005 The hierarchical atlas. *IEEE Trans. Robot.* 21: 473–481

Lourakis M I A and Antonis A A 2005 Is levenberg-marquardt the most efficient optimization algorithm for implementing bundle adjustment? In: *International Conference on Computer Vision (ICCV)*, vol. 2, pages 1526–1531

Lowe D G 2004 Distinctive image features for scale invariant keypoints. *Int. J. Comput. Vis.* 60(2): 91–110

Lu F and Milios E 1997 Globally consistent range scan alignment for environmental mapping. *Autonom. Robot.* 4(4): 333–349

Makarenko A, Williams S B, Bourgoult F and Durrant-Whyte F 2002 An experiment in integrated exploration. In: *Proceedings of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, vol. 1, pages 534–539, Lausann, Switzerland

Malik J and Shi J 2000 Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* 22(8): 888–905

Marder-Eppstein E, Konolige K and Marthi B 2011 Navigation in hybrid metric-topological maps. In: *Proceedings of International Conference on Robotics and Automation*, pages 3041–3047, Shanghai

Marinakis D and Dudek G 2010 Pure topological mapping in mobile robotics. *IEEE Trans. Robot.* 26(6): 1051–1064

McClelland M, Campbell M and Estlin T 2013 Qualitative relational mapping for planetary rovers. In: *Proceedings of Intelligent Robotic Systems, AAAI*, pages 110–113

Meyer-Delius D, Hess J, Grisetti G and Burgard W 2010 Temporary maps for robust localization in semi-static environments. In: *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5750–5755, Taipei, Taiwan

Milford M J and Wyeth G 2008 Mapping a suburb with a single camera using a biologically inspired SLAM System. In: *IEEE Transactions on Robotics Special Issue on Visual SLAM*, vol. 24(5), pages 1038–1053

Modayil J, Beeson P and Kuipers B 2004 Using the topological skeleton for scalable global metri- cal map-building. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*

Montemerlo M and Thrun S 2002 Conditional particle filter for simultaneous mobile robot localization and people tracking. In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*

Montemerlo M, Thrun S, Wegbriet B and Koller D 2002 FastSLAM: A factored solution to the simultaneous localization and mapping problem. In: *Proceedings of AAAI National Conference on Artificial Intelligence*

Montremerlo M, Thrun S, Koller D and Wegbriet B 2003 Fast-SLAM2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges. In: *International Joint Conference on Artificial Intelligence*, pages 1151–1156

Motalier P and Chatila R 1989 Stochastic multisensory data fusion for mobile robot location and environmental modelling. In: *In Fifth Symposium on Robotics Research*

Murphy K 1999 Bayesian map learning in dynamic environments. In: *Proceedings of the Conference on Neural Information Processing Systems (NIPS)*, pages 1015–1021, Denver, CO, USA

Nagatani K and Choset H 1999 Towards robust sensor-based exploration by constructing reduced generalized voronoi graphs. In: *Proceddings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1687–1698, Korea

Nathan Silberman, Derek Hoiem, P K and Fergus R 2012 Indoor segmentation and support inference from RGB-D images. In: *Proc. of ECCV*, pages 746–760

Nebot E 2000 Victoria park dataset webpage, ACFR [http://www-personal.acfr.usyd.edu.au/nebot/victoria_park.htm accessed on 2-September-2014]

Newman P and Leonard J 2003 Pure range only sub-area slam. In: *Proceedings of IEEE Conference on Robotics and Automation*, pages 1921–1926

Ni K, Steedly D and Dellaert F 2007 Tectonic SAM: Exact, out-of-core, submap-based SLAM. In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Rome, Italy*, pages 1678–1685

Niera J and Tardòs J D 2001 Data association in stochastic mapping using joint compatibility test. *IEEE Trans. Robot. Autom.* 17(6): 890–897

Oberländer J, Uhl K, Zöllener J M and Dillmann R 2008 A region-based slam algorithm capturing metric, topological and semanic properties. In: *Proceedings of IEEE International Conference on Robotics and Automation*, pages 1886–1891, Pasadena, CA, USA

Olson E 2008 *Robust and efficient robotic mapping*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, USA

Olson E 2009a Real-time correlative scan matching. In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 4387—4393, Kobe, Japan. IEEE

Olson E 2009b Recognising places using spectrally clustered local matches. *Robot. Autonom. Syst.* 57(12): 1157–1172

Olson E and Agarwal P 2013 Inference on networks of mixtures of robust robot mapping. *Int. J. Robot. Res.* 32(7): 826–840

Olson E, Leonhard J and Teller J 2006 Fast iterative optimization of pose graphs with poor initial estimates. In: *Proceedings of the IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 2262–2269

Olson E, Strom J, Goeddel R, Morton R, Ranganathan P and Richardson A 2013 Exploration and mapping with autonomous robot teams. *Commun. ACM* 56(3): 62–70

OpenSLAM 2014 www.openslam.org

Pandey G, McBride J R and Eustice R M 2011 Ford campus vision and lidar data set. *Int. J. Robot. Res.* 30(13): 1543–1552

Parr R DP-SLAM download webpage., http://www.cs.duke.edu/parr. [online accessed on 01-September-2014]

Paskin M A 2003 Thin junction tree filters for simultaneous localization and mapping In: Gottlob, G, Walsh, T (eds) *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence (IJCAI-03)*, pages 1157–1164, San Francisco, CA. Morgan Kaufmann Publishers

Paul R and Newman P 2010 FAB-MAP 3D: Topological Mapping with Spatial and Visual Appearance. In: *Proceedings IEEE International Conference on Robotics and Automation (ICRA'10)*, pages 2649–2656, Anchorage, Alaska,

Paz L, Tardos J and Neira J 2008 Divide and conquer: EKF SLAM in $O(n)$. *Robotics, IEEE Trans.* 24(5): 1107–1120

Peynot T, Scheding S and Terho S 2010 The marulan data sets: Multi-sensor perception in a natural environment with challenging conditions. *Int. J. Robot. Res.* 29(13): 1602–1607

Pfingsthorn M and Birk A 2012 Simultaneous localization and mapping (SLAM) with multimodal probability distribution. *Int. J. Robot. Res.* 32: 143–171

Pronobis A and Caputo B 2009 Cold: The cosy localization database. *Int. J. Robot. Res.* 28(5): 588–594

Ramdev R K, Krishna K M and Jawahar C V 2013 Multibody vslam with relative scale solution for curvilinear motion reconstruction. In: *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, pages 5732–5739, Karlsrule, Germany

Ranganathan A and Dellart F 2011 Online probabilistic topological maps. *Int. J. Robot. Res.* 30(6): 755–771

Ranganathan A, Kaess M and Dellaert F 2007 *Loopy SAM*. pages 2191–2196, Hyderabad; India

Ranganathan A, Menegatti E and Dellart F 2006 Bayesian inference in space of topological maps. *IEEE Trans. Robot.* 22(1): 92–107

RAWSEEDS 2009 Robotics advances through webpublishing of senorial and elaborated data sets (project FP6-IST-045144). [http://www.rawseeds.org/rs/datasets online accessed on 02-September-2014]

Reid R and Braünl T 2011 Large-scale Multi-robot Mapping in MAGIC 2010. In: *Proceedings of the IEEE 5th International Conference on Robotics, Automation and Mechatronics (RAM)*, city

Remolina E 2001 *A logical account of causal and topological maps.* PhD thesis, University of Texas, Austin

Rizzini D L and Caselli S 2010 A distributed maximum likelihood algorithm for multi-robot mapping, In: *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 573–578, Taipei, Taiwan

ROS 2014 Robot open source (ROS). www.ros.org, accessed on 02-September-2014

Ros G, Sappa A D, Ponsa D and Lopez A M 2012 Visual slam for driverless cars: A brief survey. In: *Proceedings of the 2012 IEEE Intelligent Vehicles Symposium Workshops*

Rosen D M, Kaess M and Leonard J J 2012 An incremental trust-region method for robust online sparse least-square estimation. In: *Proceedings of IEEE International Conference on Robotics and Automation*, pages 1262–1269, RiverCentre, Saint Paul, Minnesota, USA

Rosen D M, Kaess M and Leonard J J 2013 Robust incremental online inference over sparse factor graphs: Beyond the Gaussian case. In: *Proceedings of IEEE International Conference on Robotics and Automation*, pages 1025–1032

Saeedi S, Paull L, Trentini M and Li H 2011 Neural network-based multiple robot simultaneous localization and mapping. In: *IEEE Transactions on Neural Networks*, vol. 22, pages 2376–2387

Savelli F 2005 *Topological mapping of ambiguous space: Combining qualitative biases and metrical information*. PhD thesis, University of Texas, Austin

Shao L, Huang S and Dissanayake G 2013 Linear SLAM: A linear solution to the feature-based and pose graph SLAM based on submap joining. In: *Proceedings of the IEEE/RSJ Intl. Conf. on Intelligent Robotics and Systems (IROS)*, pages 24–30, Tokyo, Big Sight, Japan

Shi J and Tomasi C 1994 Good features to track. In: *IEEE Conference on Computer Vision and Pattern Recognition*, pages 593–600

Smith M, Baldwin I, Churchill W, Paul R and Newman P 2009 The new college vision and laser data set. *Int. J. Robot. Res.* 28(5): 595–599

Stachniss C 2009 *Robot mapping and exploration*. Springer-Verlag

Stachniss C and Bougard W 2005 Mobile robot mapping and localization in non-static environments. In: *Proceedings of the National Conference on Artificial Intelligence*, Pittsburgh, PA, USA

Stachniss C, Frese U and Grisetti G 2014 OpenSLAM: Open-source implementation of slam algorithms. [www.openslam.org accessed on 2-September-2014]

Steux B and Hamzaoui O E 2010 tinySLAM: A SLAM Algorithm in less than 200 line of C code. In: *Proceedings of the International Conference on Control, Automation, Robotics and Vision (ICARCV)*, pages 1975–1979, Singapore

Strassdat H, Montiel J M M and Davison A J 2012 Visual SLAM: Why filters. *J. Image Vis. Comput.* 30(2): 65–77

Sünderhauf N 2012 *Robust optimization for simultaneous localization and mapping*, PhD thesis, Technische Universität Chemnitz

Sünderhauf N and Protzel P 2012 Switchable constraints for robust pose graph. In: *Proceedings of the IEEE Conf. on Intelligent Robots and Systems*

Sünderhauf N and Protzel P 2013 Switchable Constraints vs Max-Mixture Models vs RRR - A Comparision of Three Approaches to Robust Pose Graph SLAM. In: *Proceedings of the IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 5198–5203, Karlsruhe, Germany

Tao T, Tully S, Kantor G and Choset H 2011 Incremental Construction of the Saturated-GVG for Multi-Hypothesis Topological SLAM. In: *Proceedings of the IEEE Conference on Robotics and Automation*, pages 3072–3077, Shanghai, China

Tapus A 2005 *Topological SLAM - Simultaneous Localization and Mapping with Fingerprints of Places*. PhD thesis, École Polytechnique Fédérale De Lausanne (EPFL)

Tardos J D, Neira J, Newman P and Leonard J J 2002 Robust mapping and localization in indoor environments using sonar data. *Int. J. Robot. Res.* 21: 311–330

Thrun S 2002a Particle filters in robotics. In: *Proceedings of the 17th Annual Conference on Uncertainty in AI (UAI)*

Thrun S 2002b Robotic mapping: A survey. In: Lakemeyer, G and Nebel, B (editors) *Exploring Artificial Intelligence in the New Millenium*. Morgan Kaufmann

Thrun S, Fox D and Bougard W 2006 *Probabilistic robotics*. The MIT Press

Thrun S, Gutmann S, Fox D, Burgard W and Kuipers B 1998 Integrating topological and metric maps for mobile robot navigation. In: *Proceedings of National Conference on Artifical intelligence(AAAI)*, Wisconsin

Thrun S and Leonard J J 2008 *Springer handbook on robotics*. Springer

Thrun S, Liu Y, Ng A Y, Ghahramani Z and Durrant-Whyte H 2004 Simultaneous localization and mapping with sparse filters. *Int. J. Robot. ResInt. J. Robot. Res.* 23(7): 693–716

Thrun S and Montemerlo M 2006 The graph slam algorithm with application to large scale mapping of urban structures. *Int. J. Robot. Res.* 25(5-6): 403–429

Tipaldi G D and Arras K O 2010 FLIRT – Interest region for 2D laser data. In: *IEEE International Conference on Robotics and Automation*

Tomasi C and Kannade T 1991 *Detection and tracking of point features*. Technical Report CMU-CS-91-132, Canegie Mellon Univerisy

Tomatis N, Nourbaksh I and Siegwart R 2002 Hybrid simultaneous localization and map building: Closing the loop with multi-hypotheses tracking. In: *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, pages 2749–2754, Wahington, D.C

Tong C, Gingras D, Larose K, Barfoot T D and Dupuis E 2013 The canadian planetary emulation terrain 3D mapping dataset. *Int. J. Robot. Res.* 32(4): 389–395

Tully S, Kantor G and Choset H 2012 A unified Bayesian framework for global localization and SLAM in hybrid metrictopological maps. *Int. J. Robot. Res.* 3(3): 271–288

Ultsch A and Siemon H P 1990 Kononen's self organizing maps for exploratory data analysis. In: *Proceedings of Intenational Neural Network Conference (INNC-90)*, pages 305–308 Paris, France

Walcott-Bryant A, Kaess M, Johannsson H and Leonard J J 2012 In: *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1871–1878, Vilamoura, Algarve, Portugal

Walter M, Eustice R and Leonard J 2005 A provably consistent method for imposing exact sparsity in feature-based SLAM information filters. In: *Proceedings of the International Symposium of Robotics Research (ISRR)*, pages 214–234, San Francisco, CA. Springer

Walter M R, Eustice R M and Leonard J J 2007 Exactly sparse extended information filter for feature-based SLAM. In: *International Journal of Robotics Research*, vol. 26, pages 335–359

Wang C-C, Duggins D, Gowdy J, Kozar J, MacLachlan R, Mertz C, Suppe A and Thorpe C 2004 Navlab slammot datasets, *Carnegie Mellon University*, www.cs.cmu.edu/~bobwang/datasets.html. [online accessed on 04-September-2014]

Wang C C, Thorpe C and Thrun S 2002 Simultaneous localization and mapping with detection and tracking of moving objects. In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, vol. 1, pages 842–849

Wang Z, Huang S and Dissanayake G 2006 *Implementation Issues and Experimental Evaluation of D-SLAM*, volume 25 of *Springer Tracts in Advanced Robotics*. Springer-Verlag

Wang Z, Huang S and Dissanayake G 2007a D-SLAM: A decoupled solution to simultaneous localization and mapping. *Int. J. Robot. Res.* 26(2): 187–204

Wang Z, Huang S and Dissanayake G 2007b Multi-robot simultaneous localization and mapping using d-slam framework. In: *Proceedings of 3rd International Conference on Intelligent Sensors, Sensor Networks and Information (ISSNIP-2007)*, pages 317–322, Melbourne

Wolf D F and Sukhatme G S 2003 Towards mapping dynamic environments. In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*

Wolf D F and Sukhatme G S 2004 Online simultaneous localization and mapping in dynamic environments. In: *Proceedings of the 2004 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1301–1307, New Orleans, LA

Wurm K. M., Hornung A., Bennewitz M., Stachniss C. and Burgard W. 2010 OctoMap: A probabilistic, flexible, and compact 3D map representation for robotic systems. In: *Proceedings of the ICRA 2010 Workshop on Best Practice in 3D Perception and Modeling for Mobile Manipulation*, Anchorage, AK, USA. Software available at http://octomap.sf.net/

Yang S, Wang C and Thorpe C 2011 The annotated laser data set for navigation in urban areas. *Int. J. Robot. Res.* 30(9): 1095–1099

Zhao H, Chiba M, Shibasaki R, Shao X, Cui J and Zha H 2008 Slam in a dynamic large outdoor environment using a laser scanner. In: *Proceedings of IEEE International Conference on Robotics and Automation*, pages 1455–1462, Pasadena, CA, USA

Zimmer U 2000 Embedding local metrical map patches in a globally consistent topological map. In: *Proceedings of International Symposium on Underwater Technology (UT)*, pages 301–305

Zou D and Tan P 2013 CoSLAM: Collaborative visual slam in dynamic environments. *IEEE Trans. Pattern Anal. Mach. Intell.* 35(2): 354–366