Probabilistic polygonal mesh for 3D SLAM*

*Louis-Kenzo CAHIER, Toru TAKAHASHI, Testsuya OGATA, Hiroshi G. OKUNO
Graduate School of Informatics, Kyoto University

Abstract— Autonomous localization of robotic systems and mapping are symbiotic problems that must be attacked together. This problem, known as SLAM, has advanced to the point it is considered solved. However, this is only true when mapping the environment as an auxiliary task during navigation. Dense, rich maps are needed to perform many tasks in autonomous robots, such as interaction with complex environments. We propose an untested approach based on a novel three-dimensional polygonal mesh representation of the environment that encodes probabilistic belief of the system, used as the base for a Rao-Blackwellized particle filter to perform 3D 6DOF SLAM.

Key Words: 3D SLAM, 6DOF SLAM, polygonal mesh, dense map, Rao-Blackwellized particle filter, time-of-flight camera, sound SLAM

1. Introduction

Robots have succeeded admirably in our factories; one may say they invaded them. We cannot say the same of our houses and offices. This is the next frontier: getting robots from controlled factories into unconstrained environments, and most importantly in our daily lives. Applications include daily tasks such as cleaning and cooking, medical assistance for low-mobility people, augmented reality in games and visualizations, and social roles such as companion robots.

Of particular interest to us is telepresence, *i.e.* robotic embodiement of humans in places they are not physically present, and maybe cannot be. This includes tele-work in offices, emergency response on earthquake disaster sites, or cleanup of ocean-deep drilling accidents. Incidentally, the attractiveness of robotic telepresence strongly depends on the intuitiveness of the human interface, and we believe third-person view of the remote robot should be an important element of such an interface [1].

The capacity of robots to build 3D maps of arbitrary environments through their senses is key to unlocking those promising applications. One of the most difficult challenges in achieving this is that estimating the environment's geometry from sensor data requires knowledge of the robot's position in space. In turn, a robot has no general way to know its position in space in unconstrained environments, and thus has to estimate its position by relating sensor data to its internal knowledge.

The mutual dependence of those problems gave rise, more than 20 years ago [2], to a rich field called Simultaneous Localization and Mapping (SLAM). In this timespan, probabilistic techniques based on the Extended Kalman Filter (EKF) and the Particle Filter (PF) have mostly solved SLAM, allowing recursive estimation of the robot's trajectory from sequential sensor data.

However, this indeniable success focused on solving the problem with an open choice of map type. We believe that rich, dense 3D maps are necessary, and thus feel the need to examine the SLAM problem under this constraint (discussed in section 2). Specifically, we propose exploring a new class of map representations, polygonal meshes with embedded probabilistic information, as the fundamental representation of the environment (discussed in section 3). As with any type of map representation, accompanying probabilistic inference methods have to be explored that can operate on the the probabilistic information stored in the model, as well as the maintenance of probabilistic belief in the map (discussed in section 4).

2. Properties of environment maps

The essence of SLAM is to incrementally obtain localization of the robot and build a map of its environment. Any approach thus has to choose internal representations for those two quantities. Localization is fairly unambiguous, and choice consists mainly in either restricting the robot localization space to a 2D plane using 3-dimensional position-bearing vectors $(x, y, \theta)^T$ (2D SLAM), or allowing the full range of space using 6-dimensional position-orientation vectors $(x, y, z, \alpha, \beta, \gamma)^T$ (3D SLAM). We are from now on concerned with the latter form.

2·1 3D environments

Map representations, however, have much more diversity because of the underlying complexity of the environment. Probably the most common and successful way is to model only a finite subset of points of the environment, called landmarks or features (*e.g.* distinctive visual patterns), carefully chosen so as to be easily identifiable in subsequent sensor data. The position of all features can then be estimated recursively from sensor data, along with (un)certainty of those estimates, using an Extended Kalman Filter (EKF) fed with new observations of recognized landmarks.

*This research is partially supported by JST SICP (Japan-France) on BINAAHR.

This approach, called EKF-SLAM, is the reference approach in the field.

The discrete nature of feature-maps illustrates the concept of sparse representation of the environment. By opposition, dense representations are defined as piecewise-continuous models of the environment. As we move towards more ambitious implementations of SLAM [3], many tasks that use the internal representation built by the SLAM layer, such as motion planning [4], collision avoidance, or 3D object segmentation [5], require or benefit from dense representations.

Another popular map type, called Occupancy Grid Maps (OGM) [6], divides space in cells along fixed boundaries – typically a square lattice; for each cell, the probability that it is occupied is stored in the map, and maintained by recursively processing sensor data using Bayesian filters. The size of cells is generally set to a sensible tradeoff value that makes the probabilistic update tractable while keeping as much detail as possible. While theoretically able to model any environment, the very nature of regular cell grids forces OGMs to require excessively fine cell size to model slanted surfaces with reasonable precision, as well as to represent details at multiple scales.

Finally, a popular third approach to 3D SLAM consists in incrementally registering point clouds returned by range sensors at various times into a global point-cloud map[7], using the Iterative Closest Point (ICP) algorithm [8]. This type of approach points at the possibility to explicitly modelling map uncertainty or not. Incremental maximum likelihood approaches that only keep best estimates (mode of the probability distribution) in the map cannot perform as well as those that maintin probability distributions over the map, but come at a lower computational cost. The global point-cloud map is sometimes turned into a mesh as an offline post-processing step [9]. In this case, the data is not completely synthesized into a usable dense representation online.

2·2 A new 3D map representation

With those properties of map representation in mind, we propose using a new type of map representations called "Probabilistic Polygonal Mesh" (PPM), consisting of a polygonal mesh with embedded probabilistic uncertainty. As we will elaborate in following sections, we believe such maps would present many of the advantageous properties of other representations, including:

- Provides a dense representation of geometry suitable for motion planning and physical simulation.
- Adapts to the local scale and complexity of the environment with parametrizable level of detail.
- Stores the environment efficiently in memory.
- Uses and merges all available information.
- Facilitates live visualization and exploration of reconstructed environment for human interfaces.
- Mitigates the gravity of data association errors by providing a continuous data association space.

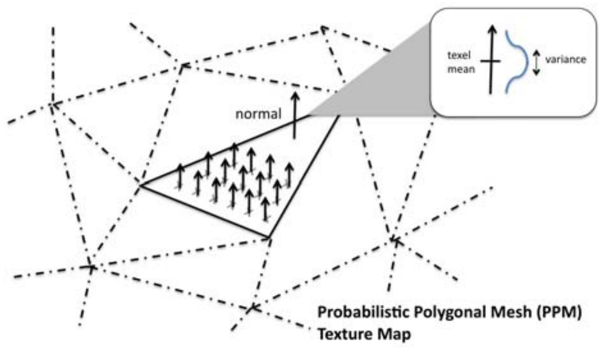

Fig.1 Embedding of probabilistic texture

The main disadvantages of using such a map representation would be the high complexity of an implementation, as well as the computational cost of maintaining such probabilistic meshes.

3. Probabilistic polygonal mesh

The fundamental idea of our contribution is to:

- Model the environment using a polygonal mesh.
- Embed probabilistic belief into this mesh to represent uncertainty.

To this end, we define a probabilistic polygonal mesh as a triplet $\mathcal{M} = (\mathcal{V}, \mathcal{F}, \mathcal{T})$ where:

- $\mathcal{V} = (v_1, \ldots, v_V) \in \left(\mathbb{R}^3 \times \mathbb{R}^3\right)^V$ is a tuple of $V = |\mathcal{V}|$ vertices; each vertex is a pair containing: a position and a normal vector.
- $\mathcal{F} = (f_1, \ldots, f_F) \in \left(\mathbb{N}_+^3 \times (\mathbb{R}_+^2)^3\right)^F$ is a tuple of $F = |\mathcal{F}|$ faces; each face is a pair containing: a triplet indicating the indices of its 3 vertices, and another triplet indicating the UV coordinates of each of the vertices.
- $\mathcal{T} \in [0 : T_x] \times [0 : T_y] \to \mathbb{R}^2$ is a 2-channel texture.

The probabilistic information stored in the mesh is located in elements of $\mathcal{T}$, called texels. The first and second channels should be interpreted as the mean and variance of a Gaussian along the normal to the face respectively (see Figure 1). This probabilistic texture is destined to receive information about the observed distribution of errors between the triangle surface and actual surface.

We now turn to describing how one might use probabilistic polygonal meshes in the context of recursive 3D SLAM.

4. Probabilistic mesh and 3D SLAM

Our proposed SLAM system consists of a Rao-Blackwellized Particle Filter (RBPF) inspired by the FastSLAM algorithm [10], adapted to work with a PPM, taking sequential 2.5D range images such as those produced by Time-of-Flight Cameras (ToFC).

A particle filter is a statistical sampling filter that maintains a constant population of particles. Each particle carries its own hypothesis for the whole trajectory robot, from the initial time, along with a probabilistic polygonal mesh map that results from the incremental map building process along the succesive past hypothesized positions (see subsection 4·3).

At each step, each particle first performs a motion update, that samples a new position from a the probability distribution given by a predetermined motion model (see subsection 4·1). Those projected particles then go through a weighting process, that rate them according to the likelihood of the sensor data given their new projected pose. Finally, particles are stochastically resampled according to their estimated likelihood, through a process called *drawing with replacement* that draws a new set of particles by repeatedly copying one of the weighted particles, selected with odds proportional to its weight.

The choice of a particle filter is motivated by the very high number of probabilistic elements in our system, along with a relatively low cost for measurement weighting (see subsection 4·2). Indeed, RBPFs have been known to support hundreds of thousands of elements with hundreds of particles [11]. This radical scaling capability is made possible by the conditional independence of probabilistic map elements when the trajectory is given, as opposed to the EKF estimation of massive covariance matrices quadratic in the number of probabilistic elements.

Initially, the probabilistic mesh map for each particle is empty, and on the first sensor data arrival, the motion update and measurement update are bypassed, going directly to the map updating step. Subsequently, at each new sensor data arrival, the motion update is first executed, followed by the measurement update, followed by the map update.

4·1 Motion update

The motion update is the simplest step and consists in stochastically sampling a new robot position, with knowledge of its past trajectory, using a predetermined motion model [6]. This results in the addition of one new position to each of the particles' trajectory variable. The motion model can be make use of speed and acceleration recovered from the latest positions, or from sensors such as Inertial Measurement Units (IMU).

4·2 Measurement update

Next we evaluate the weight of each particle. By casting a ray for each pixel in the range image onto the onto the particle's PPM from the projected position, we obtain either an intersection point, or no intersection at all. In the case of an intersection, we can compute the measurement probability of the sensor pixel by evaluating a Gaussian with variance equal to that found at the intersection point, at the algebraic distance of the sensor point from the intersection point offset by the mean found at the intersection point (see Figure 2).

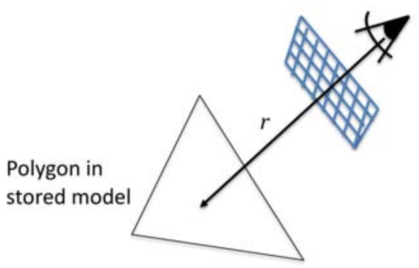

Fig.2 Measurement probability using raycasting

All pixels in the range image can then be considered independent by using the conditional independence provided by the RBPF, resulting in a the measurement probability equal to the product of the measurement probabilities of each pixel. The importance weight of each particle is its normalized measurement probability.

Use of color information When estimating the probability of a set of particles, a local environment with few distinctive geometric features will result in weakly-discriminant weights. If a visual camera is available in the robot, we can add additional visual texture chanels to the PPM, and add another term to the measurement probability that depends on matching not only the geometry, but also the visual texture in a similar way.

4·3 Map update

Every time a new range image measurement is received, it is raycast onto the PPM in the measurement step (see Figure 3). The ray-triangle intersection determined at this previous step is reused for the map updating.

Creation of new polygons If the ray doesn't intersect any face, we create a new vertex in the map at the location of the sensor data point.

Update of observed polygons If the ray does intersect an existing face, we update the intersected texture element's mean and variance to incorporate the new measurement using Bayesian inference with a zero prior mean, and prior variance as well as assumed variance tajen as the calibrated variance of the sensor [12].

Vertex adjustment Once all observations are integrated, vertices that touch texels that have been modified are reevaluated by offsetting their position along their normal by the average of all surrounding texel values.

Automatic remeshing When the mean error of a polygon is high with high certainty at a given texture element, we update the mesh to match the outstand-

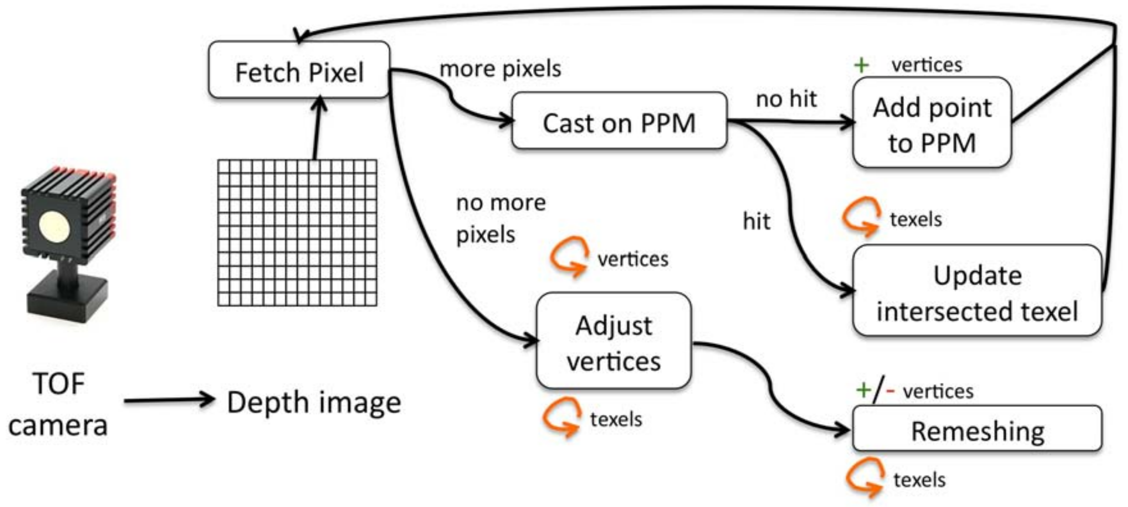

Fig.3 Processing flow diagram

ing texels by inserting a new vertex at the position, and adding faces to retriangulate the affected face.

Additionaly, we periodically run fast 3D mesh simplification on all updated faces [13].

5. On efficient implementations

As discussed above, we expect that one of the main disadvantages of PPMs is the difficulty of implementation. Many approaches to the 3D SLAM problem depend heavily on clever implementations, such as 3D OGMs implementation using space partitioning techniques. Many of the same optimizations could thus be applied to PPMs too.

Map storage Of crucial importance to PF approaches such as FastSLAM is the storage of particles in trees, and sharing common ancestry in order to avoid redundantly allocating large parts of the map that is shared by many particles.

Computer graphics Specific to PPMs is the possibility, as a mesh-based representation, to use algorithms and hardware developped for computer graphics. The measurement update step consists in large volumes of ray casting through pixels; this calls for investigating the possibility to use graphics rasterization techniques to accelerate the evaluation of particle weights.

References

[1] P. Salamin, D. Thalmann, and F. Vexo, "The benefits of third-person perspective in virtual and augmented reality?," *Proceedings of the ACM symposium on Virtual reality software and technology*, pp. 27–30, 2006.

[2] R. C. Smith, M. Self, and P. Cheeseman, "Estimating uncertain spatial relationships in robotics," in *Proceedings of the 1987 IEEE International Conference on Robotics and Automation*, pp. 850–850, Institute of Electrical and Electronics Engineers, 1987.

[3] T. Bailey and H. F. Durrant-Whyte, "Simultaneous localization and mapping (SLAM): part II State of the Art," *Robotics & Automation Magazine*, vol. 13, no. 3, pp. 108–117, 2006.

[4] J. Nieto, J. Guivant, and E. Nebot, "DenseSLAM: Simultaneous Localization and Dense Mapping," *International Journal of Robotics Research*, vol. 25, pp. 711–744, Aug. 2006.

[5] Z. C. Marton, R. B. Rusu, and M. Beetz, "On fast surface reconstruction methods for large and noisy point clouds," *2009 IEEE International Conference on Robotics and Automation*, pp. 3218–3223, May 2009.

[6] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. The MIT Press, 2003.

[7] D. Holz and S. Behnke, "Sancta Simplicitas On the efficiency and achievable results of SLAM using ICP-Based Incremental Registration," in *Proceedings of the 2010 IEEE International Conference on Robotics and Automation*, (Anchorage), pp. 1380–1387, 2010.

[8] Z. Zhang, "Iterative Point Matching for Registration of Free-Form Curves," *International Journal of Computer Vision*, vol. 13, no. 2, pp. 119–152, 1994.

[9] H. Surmann, A. Nüchter, and J. Hertzberg, "An autonomous mobile robot with a 3D laser range finder for 3D exploration and digitalization of indoor environments," *Robotics and Autonomous Systems*, vol. 45, pp. 181–198, Dec. 2003.

[10] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges," in *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*, (Acapulco, Mexico), 2003.

[11] R. Sim, M. Griffin, A. Shyr, and J. J. Little, "Scalable real-time vision-based SLAM for planetary rovers," in *Proceedings of the 2005 IEEE Intelligent RObots and Systems, Workshop on Robot Vision for Space Applications*, pp. 16–21, Citeseer, 2005.

[12] C. M. Bishop, *Pattern Recognition and Machine Learning*, vol. 248, ch. 2, pp. 67–136. Springer, Oct. 2006.

[13] P. Lindstrom and G. Turk, "Evaluation of Memoryless Simplification," *IEEE Transactions on Visualization and Computer Graphics*, vol. 5, pp. 98–115, 1999.