Scan matching in a probabilistic framework

Andrea Censi

Università degli Studi di Roma "La Sapienza"
Dipartimento di Informatica e Sistemistica "A. Ruberti"
via Eudossiana 18, I-00184 Rome, Italy
andrea.censi@dis.uniroma1.it

***Abstract*— We describe an interpretation of scan matching as a probability distribution approximation problem and we propose an algorithm that, employing a particle approximation to the target distribution, can take advantage of the knowledge of the evolution model and provide an estimate of the matching uncertainty. Experiments show it can work in unstructured environments, it is reliable to severe sensor occlusions and it handles under constrained situations gracefully.**

I. Introduction

In recent years the availability of accurate laser range finders allowed for the development of precise and reliable localization and SLAM methods. A large part of these have a "scan matching" algorithm at their core; comparing successive sensor scans improves the odometry estimate. Still, the estimate of a scan matcher has unbounded error; such error must be dealt with at an higher layer.

Several interpretations can be given to scan matching. The purely geometric interpretation is: *find a rotation* ϕ *and a translation* T *which maximize the overlapping of two groups of 2D data*, leading to a "maximum likelihood" kind of estimation. This is a general problem which has other applications outside of localization/SLAM, for example in pattern recognition.

When using scan matching for pose tracking, one usually has much more information (or, equivalently, much more assumptions) about the data. In particular, in a Bayesian framework, one assumes that the input data were generated by an underlying probability distribution, usually called *sensor model* for the exteroceptive sensors, and *evolution model* for the odometry data; both are needed to soundly estimate the posterior distribution.

For the scan matching problem one such target distribution can be $p(x_t|x_{t-1}, u_t, y_t, y_{t-1})$ (x: robot pose, u: odometry data, y: sensor data). At the very least, the uncertainty can be represented by a covariance matrix. This kind of estimate can be more useful of a MLE; for example, in SLAM methods based on global optimization [1] it can be used to change the weight and orientation of the "springs" defining the cost function; in SLAM methods based on Rao-Blackwellized particle filters [2] it can be used as an informed proposal distribution for the resampling step; it can be used to evolve a Gaussian filter.

In this paper we describe an algorithm that approximates the target distribution with a particle distribution generated with the Generalized Hough Transform (GHT). GHT Particles Matching (GPM) takes into account the evolution model and it can provide a covariance estimate. Moreover, it has the following several nice properties for a scan matcher:

- it can work in unstructured environments as it does not rely on the presence of features;
- it is robust to occlusions;
- it behaves gracefully in under constrained cases;
- it is fast: 80Hz on a desktop machine for typical input.

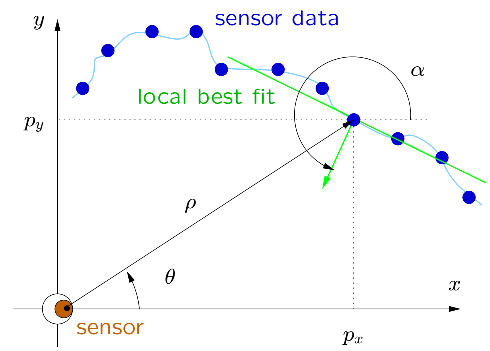


Fig. 1. θ, ρ are the polar coordinates of the sensor point; $p = (\rho\cos\theta \quad \rho\sin\theta)^t$ is the Cartesian vector; α is the surface orientation.

II. Basic algorithm

In this section we describe how to produce a particle approximation to $p(x_t|y_t, y_{t-1})$, given the two sets of points y_{t-1} and y_t (two laser scans, or a scan and a map). This is not specific to scan matching as it can be applied also to global localization; in this paper, however, we shall focus on pose tracking.

We assume that the input data are two sets of "oriented" points $y_{t-1} = \{(p_i, \alpha_i)\}$, $y_t = \{(p_j, \alpha_j)\}$, where p is the Cartesian position and α is the surface orientation (Fig. 1). Suppose that we knew that a point (p_i, α_i) in a set corresponds to a point (p_j, α_j) in the other set with a roto-translation (ϕ, T):

$$p_j = R_\phi p_i + T \quad (1)$$
$$\alpha_j = \alpha_i + \phi \quad (2)$$

In this expression, R_ϕ is a 2×2 rotation matrix. These relations can be inverted to find ϕ and T:

$$\hat{\phi} = \alpha_j - \alpha_i \tag{3}$$

$$\hat{T} = p_j - R_{\hat{\phi}} p_i \tag{4}$$

Therefore if we knew *only one* correspondence between the two sets *and* the surface orientation we could estimate both ϕ and T. The problem is that we obviously don't know such a correspondence, and that sensor data is corrupted by noise.

Previous approaches (notably the ICP [3] family of algorithms) focused on the establishment of explicit correspondences through the use of several heuristics; instead, we consider all the possible correspondences and generate all the hypotheses for ϕ and T by means of (3) and (4). This can be seen as a particularization of the Generalized Hough Transform (GHT) [4] to scan data; the crucial difference to GHT methods is that results are not accumulated in a cell array, but are considered a particle distribution. The key point is that we assume this set to be a particle approximation to $p(x|y_t, y_{t-1})$. In fact, the "correct" matchings generates dense peaks of particles around the "true" answer, while the "wrong" matchings generate random noise that can be filtered out (Fig. 2 illustrates this process).

III. Application to pose tracking

Our target distribution is

$$p(x_t|x_{t-1}, u_t, y_t, y_{t-1}) \tag{5}$$

that is we want to estimate the robot motion by comparing two successive sensor scans and taking into account the evolution model.

We assume to know the "raw" odometry and that we can evaluate the evolution model $p(x_t|x_{t-1}, u_t)$ point-wise for a fixed x_t. We also assume that the environment is not too cluttered and it has "regular" surfaces (of any shape).

The algorithm is summarized as follows:

1) Surface orientation is extracted from the scans, for example by linear regression.

2) A cloud of particles is generated from the oriented data points using (3), (4); particles not contained in a predetermined search domain are rejected.

3) Particles are assigned a weight according to the evolution model and other criteria.

4a) *First alternative:* An estimate of the mean of the distribution is computed by performing hill-climbing in this set of weighted particles.

4b) *Second alternative:* A mean and covariance estimate is computed by turning the set of particles in a set of constraints and solving a least squares problem.

A. Efficient correspondences search for radial scans

If we are given a search domain ($|T|_{\max}$ and $|\phi|_{\max}$) and one of the two input sets has a radial ordering, we do not need to consider each possible correspondence between the two sets. Consider a point (θ_i, ρ_i) on one scan; as in the other scan the points are ordered by θ_j, we can compute an upper bound on $\Delta\theta_i, = |\theta_i - \theta_j|$; we will then compare the points in the $[\theta_i - \Delta\theta_i, \theta_i + \Delta\theta_i]$ range. Intuitively, the maximum variation occurs when the point is translated by $|T|_{\max}$ perpendicular to p_i, then rotated by $|\phi|_{\max}$. Therefore $\Delta\theta_i = \tan^{-1}(|T|_{\max}/\rho_i) + |\phi|_{\max}$.

B. Particles weighting

Weighting by the motion model. We use the odometry information in two ways. First, for efficiency, we consider only the particles in a ball of fixed radius around the peak of the evolution model. Then, each particle (ϕ_k, T_k) is weighted by $p(\phi_k, T_k|x_{t-1}, u_t)$. This piece of information is necessary to get meaningful results in under constrained cases (such as a long corridor) where the sensor scans alone are not sufficient to estimate the motion (lack of observability). If the odometry information is not present, the method can still detect such cases, providing very high variance along the unobservable direction.

Weighting irregular surfaces less. We assumed that the environment surface was locally linear, therefore we weight less the particles generated by sensor points with an high residual error in the line fitting procedure.

Weighting distant readings more. A sensor scan is not an uniform sampling of the environment, as nearby objects are over-represented with respect to farther ones: a segment of $1m$ length at $1m$ distance will be intercepted by roughly an equal number of sensor rays than a segment of $2m$ length at $2m$ distance. To account for this effect, several correcting factors have been proposed [5]; one of these is to weight the particles by the distance of the sensor point, ρ_i. Of course, this is not valid if the sensor error increases with distance as it will give very high weight to semi-random observations. Note, however, that laser range finders are reliable at tens of meters distance.

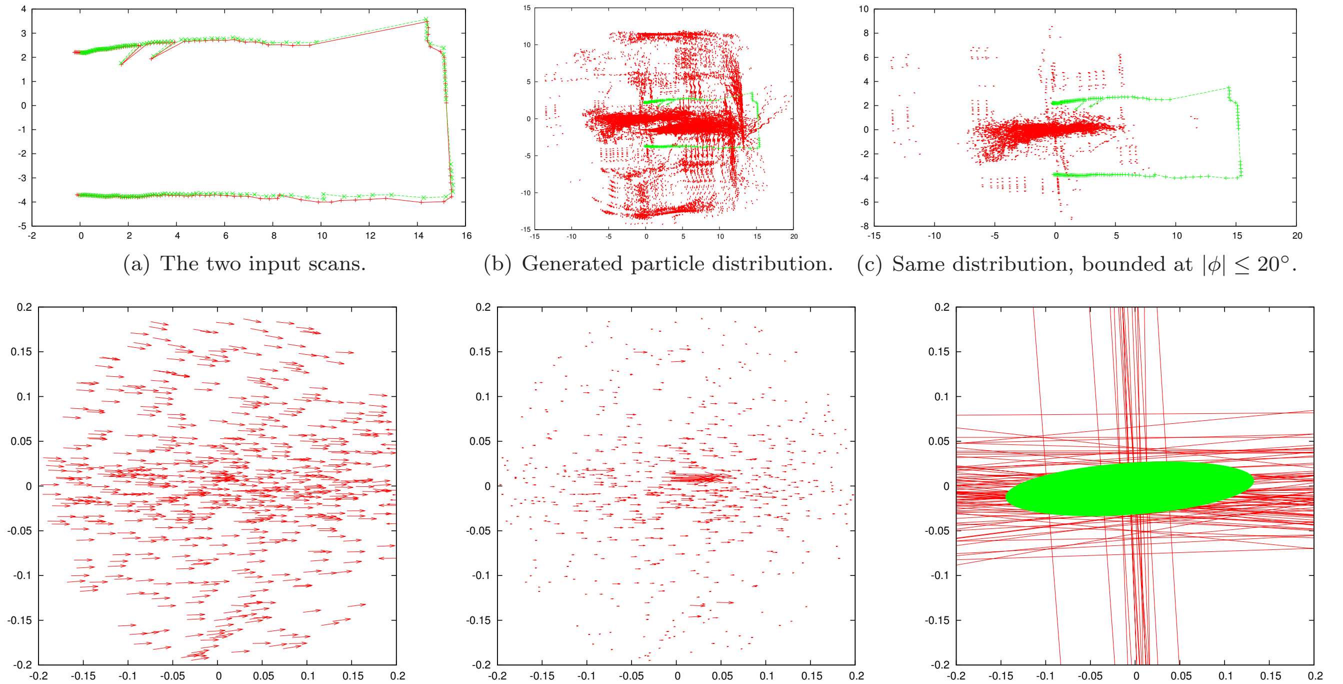
At this stage we have obtained a particle approximation to (5). The distribution can be, in general, multi modal. If one assumes that it is mono modal, the principal mode can be found by performing hill-climbing. An improvement to this basic algorithm is necessary because it is critical to characterize the uncertainty of each particle to obtain a meaningful covariance estimate. This is addressed in the next section by a simple model that turns the set of particles into a set of constraints.

IV. Linearizing the observation model

In this section we formulate a least squares (LS) problem to obtain a covariance estimate for T.

A simple model to characterize the uncertainty of (4) is to consider the information useful only along the direction α_k, the direction perpendicular to the surface. This is expressed by projecting (4) along that direction, multiplying both sides by the versor $v(\alpha_k) := (\cos\alpha_k \quad \sin\alpha_k)^t$:

$$v(\alpha_k)^t T = v(\alpha_k)^t (p_j - R_{\hat{\phi}} p_i) \tag{6}$$



(a) The two input scans. (b) Generated particle distribution. (c) Same distribution, bounded at $|\phi| \leq 20°$.

(d) Bounded at $|\theta| \leq 20°$, $|T| \leq 20cm$. (e) Same distribution after weighting. (f) Constraints with induced covariance.

Fig. 2. Fig. (a) shows the two laser scans, acquired in a corridor with irregular surfaces, aligned according to the odometry estimate. Fig. (b) shows the global distribution of particles: each particle (T, ϕ) is represented by an arrow starting at (T_x, T_y) and having direction ϕ. Fig. (d) shows the same distribution bounded at the search domain $|T| \leq 0.2m$, $|\phi| \leq 25°$. In Fig. (e) the distribution is weighted by the evolution model and the other criteria (the longer the arrow, the bigger the weight). Fig. (f) shows the constraints generated by the most weighted particles (15% of the total) and the resulting covariance of the LS estimate.

The right member consists of known terms; call it y_k. Each particle generates one constraint of the form:

$$v(\alpha_k)^t T = y_k + \sigma_k^2 \cdot \epsilon \tag{7}$$

The least squares method maximizes the likelihood of these constraints; we use as variances the inverse of the weights w_k as previously computed. We add a normalizing constant m which also models our confidence in the sensor: this will be a parameter to tune.

$$v(\alpha_k)^t T = y_k + m/w_k \cdot \epsilon \tag{8}$$

Note that it is an approximation to consider the constraints to be independent: one sensor reading produces more than one constraint and those are surely correlated.

The LS model (9) appears as follows:

$$\begin{aligned}
LT &= Y + R \cdot \epsilon && (9)\\
\chi^2 &= (LT - Y)^t R^{-1} (LT - Y) && (10)\\
\overline{T} &= \arg\min \chi^2 \\
L &= \big(v(\alpha_1) \cdots v(\alpha_k) \cdots v(\alpha_K)\big)^t && (11)\\
Y &= (y_1 \quad \ldots \quad y_k \quad \ldots \quad y_K)^t && (12)\\
R &= m \cdot \mathrm{diag}\{1/w_1, \ldots, 1/w_k \ldots 1/w_K\} && (13)\\
\overline{T} &= (L^t R^{-1} L)^{-1} L^t R^{-1} Y && (14)\\
&= \left[\sum_k w_k v(\alpha_k) v(\alpha_k)^t\right]^{-1} \sum_k w_k y_k v(\alpha_k)
\end{aligned}$$

$$\begin{aligned}
C &= (L^t R^{-1} L)^{-1} && (15)\\
&= m \left[\sum_k w_k v(\alpha_k) v(\alpha_k)^t\right]^{-1}
\end{aligned}$$

L has dimensions $K \times 2$ ($K \simeq 1000$ for typical input) therefore we must invert: the R matrix that we constructed diagonal, and a 2×2 matrix $(L^t R^{-1} L)$, always invertible if L is full rank. C is the covariance matrix: note that m relaxes or tightens the covariance while it has no effect on T. Also note that C does not depend on the actual observation Y: indeed this is correct if one assumes to know precisely the observation covariance matrix R. It is possible to take into account the goodness of fit by weighting C by the mean squared error: $\Sigma = (\chi^2/K)C$.

A. *Tuning*

Tuning is necessary because some of the weighting factors described in Section III-B do not have a precise probabilistic interpretation. We derived this expression for the covariance:

$$\Sigma = m \frac{\chi^2}{K} \left[\sum_k w_k v(\alpha_k) v(\alpha_k)^t\right]^{-1} \tag{16}$$

Note that: 1) the "shape" of the covariance matrix, that is the eigenvectors and the eigenvalues ratio, does not depend on m; 2) the ratio between eigenvalues of two successive covariance estimates does not depend on m. Therefore, even if we can't assume to have a good estimate for m, 1) we know what are the directions of ambiguity for the pose estimation; 2) we know the relative confidence for two different scan matching operations. A tuning procedure of m requires the knowledge of the ground truth and the definition of the desired level of confidence. Assume to know at each time step: x_t (the ground truth), $\overline{x}_t = (\overline{\phi}, \overline{T})$ (GPM result), $\Sigma_t(m)$ (covariance guess, dependent on m). The value m can be chosen as to $\overline{x}_t \sim \mathcal{N}(x_t, \Sigma_t(m))$ is satisfied to the desired level of confidence.

V. Experimental results

In this section we first illustrate some statistical analysis of GPM on simulated data, then we comment about one run on real data.

The main reason to do simulated experiments was the availability of the ground truth. We discarded the idea to use as ground truth for real data the output of another algorithm for these reasons: 1) *a priori*, GPM could have better results 2) there could be common errors for both algorithms 3) other SLAM algorithms would execute matchings with different frequency and this did not allow accurate comparisons.

A scan matcher needs a lot of parameters that deeply influence the results; rather than hand-tuning the algorithm for each trial, we fixed the following parameters (reasonable for robots with imprecise odometry in indoor environments) for all experiments, for both simulated and real data:

- execute scan matching every $20cm$ or $10°$;
- search domain: $|T| \leq 20cm$, $|\phi| \leq 25°$;
- for extracting the surface direction at point p_i we considered points $p_{i-3:i+3}$.

In both cases GPM was executed "open loop", that is on successive pairs of sensors scans. Conversely, building a map while matching does give better results, but the statistical analysis would have become more intricate (we should have included the mapping algorithm in the analysis).

A. Simulated experiments

We used the `player/stage` simulator[1] for these experiments, in different environments: structured (Fig. 3(a)), structured but cluttered (Fig. 3(b)), unstructured (Fig. 3(c)); these are typical cases for indoor environments. We collected a sensor log and then corrupted it with noise: we added Gaussian noise on the sensor readings and Gaussian and systematic noise on odometry. We show here only the results for sensor noise, because GPM proved to be very reliable to odometry errors (note that the evolution model influences only the weight and not the pose of the particles).

Fig. 3 shows, for increasing values of the sensor noise deviation σ, a measure of the error and a measure of GPM's estimate of its own performance. We note that the performance degrades gracefully, more or less linearly with respect to the sensor noise, even with σ as large as $8cm$. On the other hand the confidence estimate does not reflect accurately the actual error with noises higher than $3cm$, probably because the error on the extraction of the surface direction causes the failure of the model (9); obviously the effect is more marked in the cluttered environment.

σ	$1cm$		$2cm$		$4cm$	
	ϕ	T	ϕ	T	ϕ	T
structured	0.56	1.1	0.69	1.4	0.99	1.9
cluttered	0.75	1.2	0.86	1.5	1.14	2.3
unstructured	0.82	1.0	0.98	1.5	1.22	2.5

Fig. 4. Other statistics for the experiments run on the environments depicted in Fig. 3; we show here the $e(\phi)$ deviation (degrees) and the mean norm of $e(T)$ (cm).

B. One anecdote

This log, collected by a Pioneer robot in a mine[2], has several interesting characteristics, some which are hardly reproducible in a 2D simulator: unstructured environments, rough terrains, biased odometry and frequent sensor occlusions. Fig. 5(a), 5(b), 5(c) show the map, the raw sensor log and GPM's result. Fig. 5(d) shows, for each time step, the minimum and maximum eigenvalue of the covariance matrix for x, y estimated by GPM. The higher the eigenvalues, the more uncertain was GPM about its estimate. We cross-labeled some spots in Fig. 5(a) and Fig. 5(d) for easy reference.

We note the following patterns:

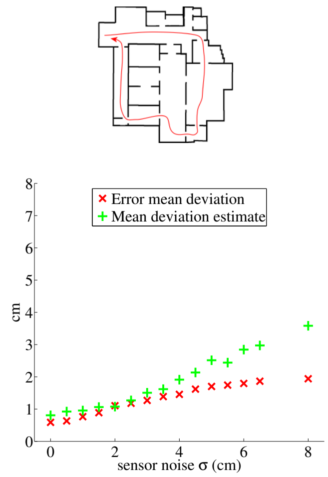
1) Define as x the direction to the front of the robot and as y the lateral direction. We found out that for a robot traveling in an environment structured with rooms and corridors, the eigenvector associated to the minimum eigenvalue lies along y, that is the uncertainty along x is almost always bigger, because usually the robot sees two lateral surfaces nearby and one frontal surface far away.

2) When traveling in a corridor, the uncertainty along x is maximum at the beginning and then decreases (see spots marked $1, 2, \ldots$).

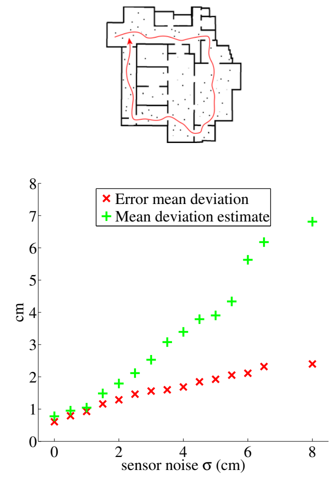
3) The occasional spike of uncertainty along x (see for example spot "B") corresponds to an instant when the robot tilted and saw the ground with the sensor. In this case GPM continued to give a good estimate along y and signaled the uncertainty along x. The area marked as "A" corresponds to a steep ascent when this phenomenon was continuously observed.

[1]`http://playerstage.sourceforge.net`

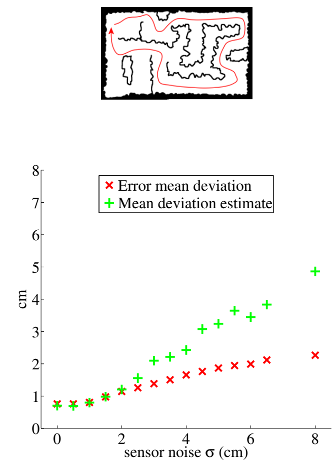
[2]Thanks to Dirk Haehnel and the CMU mine-exploring group for the data files.



(a) Structured environment

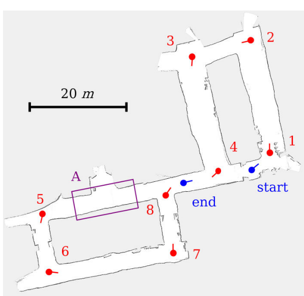


(b) Cluttered environment

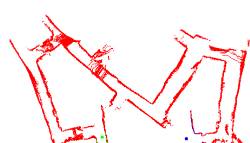


(c) Unstructured environment

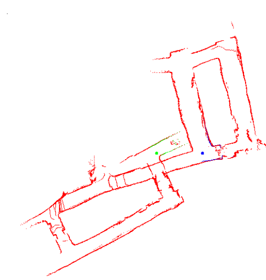
Fig. 3. Using `player/stage` we created a perfect sensor log for each of the environments. The odometry information was perturbed with i.i.d. Gaussian noise of variance proportional to the traveled distance, as to have $\sigma_x = \sigma_y = 20cm$ for a distance of $1m$ and $\sigma_\theta = 2°$ for $10°$. The range data (180 rays/180°) were perturbed with Gaussian noise of increasing deviation. GPM was executed with the same parameters mentioned in Section V. For each log, GPM has been tuned on half of the data set as to have 95% of the errors fall into a 95% confidence interval of the estimated covariance; the plotted statistics were computed on the other half. In the figures they are plotted, for increasing values of the sensor noise deviation, a measure of the covariance of the x, y error distribution ($\sqrt[4]{\det(C_T)}$, where C_T is the sample covariance over T trials) and the mean deviation of GPM's covariance guess ($\frac{1}{T}\sum_{t=1}^{T}\sqrt[4]{\det(\Sigma_t)}$). The measure considered is the geometric mean of the principal deviations: $\sqrt[4]{\det(C_T)} = \sqrt[4]{\lambda_1\lambda_2} = \sqrt{\sigma_1\sigma_2} = \overline{\sigma}$.



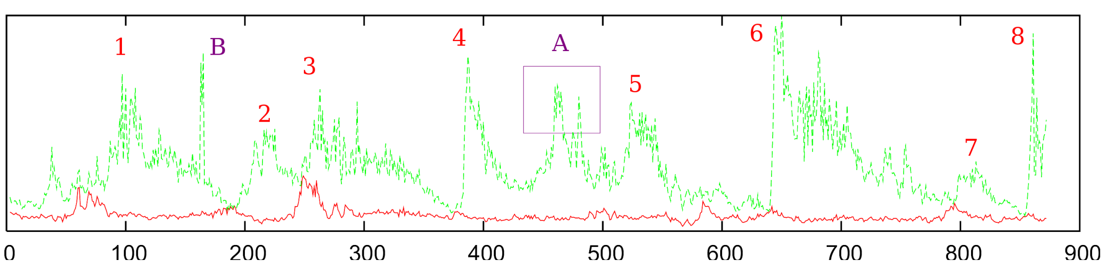
(a) Test environment



(b) Odometry data



(c) GPM scan-matching result



(d) Eigenvalues of x, y covariance matrix (algorithm in Section IV)

Fig. 5. These figures show the results of GPM on a real-world sensor log. Fig. 5(a) shows a map of the environment: the robot started at "start" and, following the numbers, traveled the two loops up to "end". Fig. 5(b) shows the odometry data (as we can see corrupted by a systematic error on ϕ). Fig. 5(c) shows the result of GPM; it has been run "open loop" on successive pairs of laser scans, without building a map. Fig. 5(d) shows, for each time step, the covariance estimate represented with the value of the minimum and maximum eigenvalue. The y-scale is not shown as the values scale linearly with m, which should be tuned. Certain time instants are cross-labeled as in Fig. 5(a): see Section V-B for a comment of the patterns in the data.

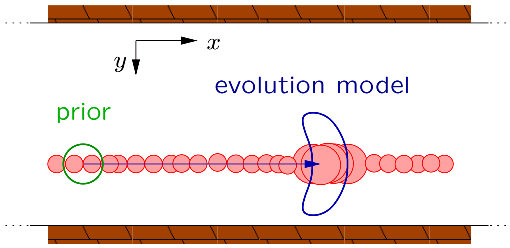


Fig. 6. This figure shows the behavior of GPM in severely under constrained situations. The observations and the geometry of the environment determine the position of the particles and the evolution model determines their weight. Therefore the y estimate depends for the most part on the sensor data and the x estimate on the odometry data.

VI. Related work

It is interesting to compare GPM to the popular ICP family of methods. There are two situations in which GPM is clearly superior. The first is the so called "infinite corridor" (Fig. 6). While ICP-like methods have convergence problems in under constrained situation, GPM would have the best possible behavior: the particles would be distributed along the corridor direction and weighted according to the evolution model. The other situation is that of occlusions, especially due to violations of the assumption of environment planarity - for example, a robot tilting and seeing the ground. ICP-like methods would converge to a false local minimum.

On the other hand, ICP-like methods, while not having global convergence properties as others [6], can compensate greater errors on ϕ than GPM. The reason is that to have a realignment of, say, $> 45°$, one should consider a search domain of $> 90°$ and the particle distribution would probably be highly multi modal and a clustering step would be needed.

To our knowledge, there exist no ICP-like methods that take advantage of the knowledge of an arbitrary explicit evolution model/prior for estimating the final uncertainty, nevertheless research has been done in this direction: [7] takes into account the evolution model (assumed Gaussian) for establishing the correspondences; assuming that the correspondences hold, one can estimate the covariance by considering the Hessian of the quadratic form being minimized [8].

We now compare GPM to two other particle-based methods to obtain an estimate of the robot motion. The first, a "degenerated" Monte Carlo Localization, has been proposed in [9]: 1) sample $\{x\}_k$ from the evolution model 2) weight the particles according to a likelihood $p(y_t|x_t, y_{t-1})$ (note that in this case it is defined the likelihood of a sensor reading with respect to another reading). The second [2] is used in a SLAM setting. 1) assume as the mean of the distribution the result of a scan matcher. 2) sample $\{x\}_k$ uniformly around the mean. 3) weight the particles by a likelihood $p(y_t|x_t, \text{map})$. (note that in SLAM the target distribution is slightly different than (5) because it is conditioned also on the map). The difference of GPM with respect to both of these is that it is a deterministic algorithm as it does not use any random sampling. Moreover, it does not need the definition and evaluation of a likelihood: evaluating the likelihood of a particle is usually expensive.

VII. Conclusion and future work

We have proposed a scan matching algorithm that provides a sound particle representation of the target distribution. This representation allows to take into account the evolution model and to handle under constrained cases gracefully. Moreover, it can work in unstructured environments and with moderate amount of sensor noise.

We have fruitfully applied the basic algorithm of GPM to global localization. As for pose tracking, future work will concern refining the current method. Detecting multi modality through clustering of the particle distribution could improve efficiency as it would allow for greater matching steps, while using a more complex model [10] to characterize the uncertainty in (4) might lead to higher accuracy.

Acknowledgment

We would like to thank Jens-Steffen Gutmann and the anonymous reviewers for insightful comments on the draft of this paper.

References

[1] F. Lu and E. Milios, "Globally consistent range scan alignment for environment mapping," 1997.

[2] G. Grisetti, C. Stachniss, and W. Burgard, "Improving grid-based slam with rao-blackwellized particle filters by adaptive proposals and selective resampling," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2005.

[3] F. Lu and E. Milios, "Robot pose estimation in unknown environments by matching 2d range scans," in *CVPR94*, 1994, pp. 935–938.

[4] D. H. Ballard, "Generalizing the hough transform to detect arbitrary shapes," in *Readings in computer vision: issues, problems, principles, and paradigms.* San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1987, pp. 714–725.

[5] J. Forsberg, U. Larsson, and øAke Wernersson, "Mobile robot navigation using the range-weighted hough transform," *IEEE Robotics & Automation Magazine*, vol. 2, no. 1, pp. 18–26, Mar. 1995.

[6] A. Censi, L. Iocchi, and G. Grisetti, "Scan matching in the hough domain," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'05)*, 2005. [Online]. Available: http://www.dis.uniroma1.it/∼censi/

[7] L. Montesano, J. Minguez, and L. Montano, "Probabilistic scan matching for motion estimation in unstructured environments," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'05)*, 2005.

[8] O. Bengtsson and A.-J. Baerveldt, "Robot localization based on scan-matching - estimating the covariance matrix for the idc algorithm," in *Journal of Robotics & Autonomous Systems*, 2003, pp. 761–768.

[9] D. Silver, D. Bradley, and S. Thayer, "Scan matching for flooded subterranean voids," in *IEEE conference on Robotics Automation and Mechatronics (RAM)*, December 2004.

[10] S. T. Pfister, K. L. Kreichbaum, R. S. I., and J. W. Burdick, "Weighted range sensor matching algorithms for mobile robot displacement estimation," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, Washington, DC, May 2002.