Proceedings of the 2006 IEEE International Conference on Robotics and Automation
Orlando, Florida - May 2006

Extended EM for Planar Approximation of 3D Data

Rolf Lakaemper
Computer and Information Sciences Dept.
Temple University
Philadelphia, PA, 19122, USA
Email: lakamper@temple.edu

Longin Jan Latecki
Computer and Information Sciences Dept.
Temple University
Philadelphia, PA, 19122, USA
Email: latecki@temple.edu

Abstract – **The paper deals with fitting of planar patches to 3D laser range data obtained by a mobile robot. The number and the initial position of the patches are unknown, hence their estimation is a challenging problem. It is solved by adding iterated steps of split and merge to a modified Expectation Maximization (EM) algorithm. This allows for precise adjustment of the number of patches, independent from the initial model. The proposed approach overcomes the problem of classical EM, which produces an optimal solution only if the number and position of model components is well estimated.**

Index Terms - 3D Robot Mapping, EM

I. INTRODUCTION

Our domain of interest is approximation of 3D data points using planar patches. Existing solutions make assumptions about the number of fitted patches, extend of noise, and/or the order of data points. We do not make any assumptions about the order of data points and extent of noise. On contrary to the existing approaches, the proposed method avoids the problem of a locally optimal solution and produces stable approximations to 3D datasets. Moreover, the final number of fitted patches is not pre determined but depends on the objects represented by the data and the extent of noise in the data. This means that the number of model components is adjusted to achieve the best possible approximation accuracy as a function of noise extent. The proposed approach adds two new steps to EM that are well and intuitively integrated with the standard E and M steps. In the first new step, the model components (i.e. the patches) obtained by a previous EM iteration are examined for support of the data points. The main idea is that a higher and homogeneous point density around a patch indicates a presence of a linear structure in the data points. Parts of the components that do not have sufficient support are removed, leading to component splitting and removal. This results in a new set of model components for the next EM iteration.
The second new step is merging similar model components. It prevents generating statistical models that overfit the data, i.e., fit noise in the data. This step requires a similarity measure of patches. Our approach bases the similarity on principles of perceptual grouping used to merge pairs of patches, visually belonging together, to a single patch. Perceptual grouping is rooted in human perception and is an active research topic in computer vision It dates back to the German school of Gestalt psychology in the beginning of 20th century [10].

Since the similarity measure of model components requires domain specific knowledge, we present the proposed methodology in a context of a particular domain. However, the proposed framework provides a domain independent extension of EM.

Assuming that the initial number of model components is well estimated, the main difficulty of fitting planes or patches to point data is that the correspondence of data points to them is unknown. The Expectation Maximization (EM) algorithm presented in [2] provides an iterative solution to the correspondence problem (in fact, EM applied to (2D) line fitting is known as the Healy-Westmacott procedure in statistics [3], and predates EM by many years). Our departing point is a 3D EM plane-fitting algorithm. However, since our goal is to fit patches, not planes, to point data, we trim planes to patches, which is a simple extension of EM (Section II.A).

The main contribution of this paper is the introduction of non-reversible split and merge steps. Both split and merge require only local evaluation. Due to the integration of these operations in the EM framework, we are able to obtain a globally optimal solution after just a few iterations (between 5 and 15) in all our experiments.
Other 3D mapping approaches, including Hough transformation, grid based and EM based algorithms can be found in [4],[6],[7],[9],[11],[12]. An overview of approaches to obtain polygonal 2D maps from laser range data can be found in [8].
The proposed approach provides a solution to the well-known problem of local optimum in EM. A classical case of an EM local optimum problem is illustrated in Fig. 4 . Iterative steps and results of our approach on generated (ground truth) and real datasets and are found in section III.

II. SPLIT AND MERGE IN THE EM FRAMEWORK

We begin with a short overview of EM applied to plane fitting. The core of the procedure is simple least square fitting which is dimension independent, hence plane fitting is the 3D version of line fitting, being presented in detail in [1]. 3D EM fits (infinite) planes to the data given. Taking a set of data points in 3D space and an initial set of 2D planes as input, the algorithm alternates the following two steps until it converges

(the algorithm is guaranteed to converge to some *local* optimum).

- **E-Step** (Expectation Step): Given a current set of planes, for each point the probabilities of its correspondences to all planes are estimated based on its distances to planes.
- **M-step** (Maximization Step): Given the probabilities computed in the E-step, the new positions of the planes are computed using a regression weighted with these probabilities.

A. EXPECTATION MAXIMIZATION PATCH FITTING (EMPF)

The data structure we utilize to fit the data is a set of planar rectangles, called *patches*, which are subsets of the planes computed by the general EM. To be more versatile, each patch is subdivided into a grid of *tiles* (see fig.1). We distinguish between *supported* and *unsupported* tiles by the number of data points close to the respective tile (more detailed explanation below). All computations are solely processed on the set of supported tiles, e.g. the distance of a point to a patch is the distance of the point to the closest supported tile, see fig.1.
The *grid size G* of a patch defines the length of the edges of its tiles (boundary tiles are resized to fit the patch as necessary, see fig. 1). The grid size is an important steering parameter of the modified EM. Its computation and role in the system is explained in further detail in section II.F.
To conclude the data structure setup: the final result of the modified EM is a set of patches identifying planar macro structures (e.g. walls) that consist of a collection of (coplanar) supported tiles. The tiles identify the shape of these structures in a granularity or resolution determined by the patch's grid size.

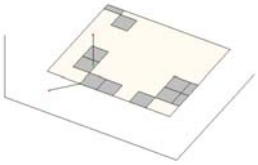

Fig. 1. Patch with supported tiles. The outer rectangle is the patch, the dark inside rectangles are supported tiles. The unsupported tiles, completing the grid, are not shown. All tiles are of same size, with the exception of boundary tiles, being resized to fit the patch. The distance of the two example data points (red) to the patch is the min. distance to the set of supported tiles, as indicated by the connections between data points and the closest points (green) on the tiles.

The proposed approach requires a minor extension of the general EM plane fitting to work with patches, which we will call **Expectation Maximization Patch Fitting (EMPF)**.
The only difference of EMPF in comparison to EM plane fitting is, that the input/output consist of patches, instead of infinite planes. The input additionally consists of the set of data points to be fitted.
EMPF is composed of the following three steps:

1. E-step with patches (the EM probabilities are computed based on the point distances to sets of supported tiles)
2. M-step with the probabilities computed in the E-step, resulting in infinite planes containing the new patches.
3. Trimming planes to patches and determining supported tiles.

Step (3) is composed of two sub steps:

1. Assignment of supporting data points to planes.
2. Cutting the planes to patches and tiles by distance projection.

Detailed description of each step:
E-step: First we need to recall the computation of EM probabilities. Let $a_1,..., a_m$ be a set of data points in 3D space, and let $s_1,...,s_n$ be a set of patches. Usually m is significantly larger than n. For each point a_i, the probability p_{ij} that a_i corresponds to patch s_j is computed for $j=1..n$.
Formally, $p_{ij} = p(z_i=j)$, where z_i is the hidden variable associated with point a_i whose values range over the patch indices. Analog to EM plane fitting, this probability is computed based on the distance $d(a_i,s_j)$ from point a_i to patch s_j, i.e. the distance to the closest supported tile in patch s_j:

$$p_{ij} \sim exp(- (d(a_i,s_j)^2 / 2S^2) \quad (1)$$

and normalized so that $\sum_{j=1..n} p_{ij} = 1$ for each i.
The standard deviation S in eq. (1) scales the weights p_{ij} with respect to the patch's grid size G in a way that points in distance G have a constant weight W (in our system 1/100) before normalization. This guarantees independence from the data points' scale and, since G is decreasing during the iterative EM process (see below), emphasizes the role of local support (i.e. closer) points to determine the planes' positions during the iteration. S is computed by:

$$S = G/sqrt(-2 \log(W)) \quad (2)$$

(Substituting S in (1) with the right side of (2) yields W for $d(a_i,s_j)=G$).
Therefore the two differences to the standard EM plane fitting are first the replacement of the distance point to plane with the distance point to closest supported tile in patch, and second, the use of a distance scaling factor S.
After every E-step we obtain a matrix (p_{ij}), with each row i representing the patch affiliation probabilities for point a_i, also called the *support* of point a_i for the patches s_j. Each column j can be seen a set of weights representing the influence of each point on the computation of a new patch position in the M-step.

M-step: the output of the M-step, which performs an orthogonal regression weighted with (p_{ij}), is a set of (untrimmed) planes $e_1, ... , e_n$ corresponding to the input

patches $s_1, \ldots, s_n$. The normal vector to the plane e_j is the eigenvector to the smallest eigenvalue of the matrix M_j defined as (all sums to be read as $\sum_{i=1..m}$):

$$\begin{vmatrix} \sum p_{ij}(a_{ix}-X)^2 & \sum p_{ij}(a_{ix}-X)(a_{iy}-Y) & \sum p_{ij}(a_{ix}-X)(a_{iz}-Z) \\ \sum p_{ij}(a_{iy}-Y)(a_{ix}-X) & \sum p_{ij}(a_{iy}-Y)^2 & \sum p_{ij}(a_{iy}-Y)(a_{iz}-Z) \\ \sum p_{ij}(a_{iz}-Z)(a_{ix}-X) & \sum p_{ij}(a_{iz}-Z)(a_{iy}-Y) & \sum p_{ij}(a_{iz}-Z)^2 \end{vmatrix}$$

where $a_i=(a_{ix}, a_{iy})$ are the coordinates of the data points, and (X,Y) is their average weighted with p_{ij} for $i= 1...n$. (X,Y) also defines a point on plane e_j , hence the plane e_j is uniquely defined by (X,Y) and M.

Trimming: In order to trim the planes to patches with supported tiles, we first need to assign *max. supporting* data points to planes (step 3.1). This assignment is based on the probabilities computed in the E-step.
A *support set* $S(s_j)$ for a given plane e_j is defined as set of points whose probability of supporting plane e_j is the largest (in comparison to other planes), i.e.,

$$S(e_j)=\{a_i : p_{ij} = max\ (p_{i1}, \ldots, p_{in})\ \}.$$

A point a_i *supports* a plane e_j if $a_i \in S(e_j)$.

Trimming the planes to patches (step 3.2) is a simple step now, using the support set S.
For each j, we define the set $S_P(e_j)$ as the set $S(e_j)$ projected (orthogonal) onto the plane e_j. The *trimmed patch* s_j is the minimum bounding rectangle of $S_P(e_j)$, with one edge direction defined by the principal axis of $S_P(e_j)$ .We obtain a set of patches

$$s_1, \ldots, s_n \text{ with } s_j \subset e_j.$$

The support set for each patch is simply the support set of the corresponding plane, i.e., $S(s_j) = S(e_j)$ for $j =1, \ldots, n$, a point a_i *supports* a patch s_j if $a_i \in S(s_j)$.

Determination of supported tiles:
A patch s_j is decomposed into equal tiles of edge length G (the grid size). For each tile t_k of s_j we define its support $support(t_k)$ as the number of data points supporting s_j in the cube $C(t_k)$, a cube whose edge length is G, placed around t_k as shown in fig. 2.

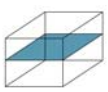

Fig. 2. Cube around tile t_k (shaded). All edges have length G (grid size).

We call the union set of points meeting these requirements for all tiles t_k of a patch s_j the *reduced* support set of s_j , $S^r(s_j) \subset S(e_j)$. see fig. 3.

In each iteration a support threshold T is computed from the statistics of the $support(t_k)$ values over all tiles of a patch. Tiles t_k with $support(t_k) > T$ are marked as supported tiles.

If a patch does not contain supported tiles, it is removed.

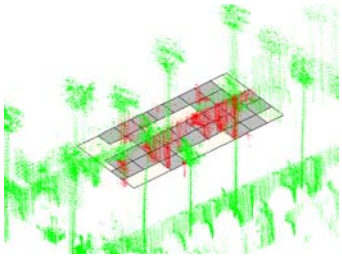

Fig. 3. Single patch with supported tiles and support points. The red data points show the reduced support set S^r of the patch. green: other data points.

The parameters G and T are computed dynamically each time in the trimming step. G is computed as $3\ std(d)$, where $std(d)$ is the standard deviation of point patch distances computed for all data points to the patches they support (see also section II.F for more information about G). T is computed as $mean(c) - 2\ std(c)$, where c is the number of points in the reduced support set S^r.
It should be noted that computing T needs to be done with the current G and not the prior. This can be achieved by recounting points in $C(t_k)$ before computing T.

B. SPLIT AND MERGE EM PATCH FITTING

The following introduces the outline of the proposed algorithm. The proposed **split and merge EM patch fitting (SMEMPF)** algorithm iterates the following steps (described in detail below):

1) EMPF (Expectation Maximization Patch Fitting)
2) PS (Patch Split): data support evaluation of patch obtained by EMPF (Section II.C)
3) EMPF
4) Patch merge (Section II.D)

We alternate patch splitting and merging between the steps of the patch fitting EM algorithm.

The main goal of PS is to evaluate the quality of the EMPF output, i.e., how well the EM positioned the new patches. Patches will be split into multiple patches based on the distribution of supported tiles. If a patch contains a large number/area of unsupported tiles, it will be split into multiple coplanar patches to allow a better fit of the supported tiles to the data in the next EMPF step (3). Hence PS creates a higher number of patches in order to optimally fit the input data points; the number of patches is *not constant*, as it is in classical EM. This way we overcome the problem of locally optimal solutions as appearing in the classical EM due to a wrongly defined number (too low) of fitting patches, since such a solution will not have a good global support in the data points. Additional to the starting number of patches the initial position of the patches also does not matter, since the following EMSF will reposition the split patches to better fit the data.

In the merging step (4), pairs of similar patches are merged to single patches. Due to this process, the number of patches cannot grow to infinity. Hence the number and position of the new patches introduced by the split is not critical in the modified EM framework. Iterating split and merge in the EM framework is a powerful tool to adjust the number and position of patches to better fit the data points.

A few iterations of the proposed algorithm are illustrated in fig. 7 and 8. The proposed algorithm converges, since EM converges and the PS procedure (Section II.C) stops splitting if a certain goodness of fit criterion is met. Usually just a few iterations of steps (1)-(4) are required, e.g. 8 iterations in the outdoor experiment (Stanford Campus, fig. 8, 9). Our stop condition is the stability of distances of data points to the closest patches. It has to be mentioned that in between certain iterations (e.g. all 5 iterations) new patches are added if there's a high number of points not supporting any patches. This situation can occur if the initial patch setup is placed far away from certain data points.

C. PS: Patch Split

A classical case of the EM local optimum problem is illustrated in fig. 4.

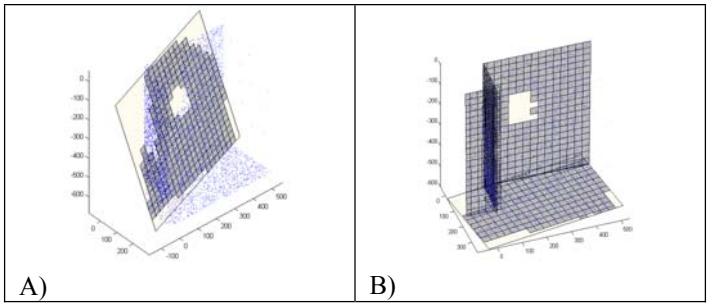

Fig. 4: Fitting data without and with splitting. A) fitting the dataset with one patch (result of classical EM with initial model of 1 patch). B) after splitting into 4 patches using SMEMPF. The ground truth model can be seen in fig. 7

Clearly, the problem in fig.4 A) is that only one patch is used, while 4 patches are needed. B) shows the result using SMEMPF, automatically gaining the required number of patches.

Fig. 5 illustrates the split operation described in this section. Splitting is processed along axes in the patch having insufficient support of data points. First the points $a_i \in S^r(s_j)$ of the reduced support set of patch s_i are projected onto the patch, yielding a point set in a 2D coordinate system defined by s_i. In this 2D system, the points a_i are projected onto the X-axis under different rotations (0,45,90,135 degree) to gain density information along the respective directions. On the X-axis, they are quantized into bins of size G, the grid size. The bin containing the minimum amount of points min_p defines position and direction of a split axis. If min_p falls below a threshold $min_p < 2/3(\#S^r(s_j))$, (# = number of points), a split is caused. The split divides $S^r(s_j)$ into two sets of points $S^{r+}(s_j)$ and $S^{r-}(s_j)$, left and right of the split axis. Two new patches s_j^+ and s_j^- are created in the plane e_j, the plane containing s_j. Using $S^{r+}(s_j)$ and $S^{r-}(s_j)$, they are trimmed and their supporting tiles are determined. s_j^+ and s_j^- then replace the original patch s_j in the model. The split procedure is recursive: s_j^+ and s_j^- again are processed the same way, until each resulting patch has a sufficiently homogeneous distribution of supporting points and is not split further.

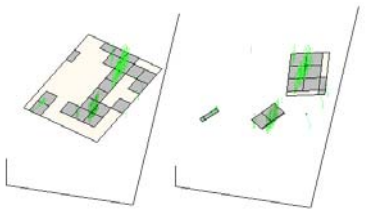

Fig. 5: Split. Left the original patch, right the split patches. The three new patches are slightly replaced compared to their origin, due to new patch directions determined by the principal axis of projected support points. Yet, by definition they still are coplanar with the original patch.

Note that the (number of) points in $S^r(s_j)$ depends on the grid size G, which therefore indirectly steers the splitting process. A smaller G enables reaction to more local density differences, and the split resolution is higher (due to smaller bin size). See section II.F for further notes on G.
The resulting new patches are coplanar with the original patch s_j, split determines a new *number* of patches, a number that leads to a better fit to the data due to the *replacement* of the newly created patches in the follow up EMPF step 3.

D. Merging

Motivation: iterated EMPF followed by PS only, without merge, could possibly grow the number of patches with each iteration to a potentially large number. Therefore, merging 'similar' patches is necessary. Merging is responsible for the accuracy of the statistical model, without it the model may end up fitting the noise. The main idea is, that if a given patch is properly split, then EMPF will reposition the resulting patches to better fit the data points in a way that will turn them away from each other. If a patch is unnecessarily split, the patches remain very similar after an EMPF iteration, where similar means that they will be nearly collinear and close to each other. This suggests that merging should combine two perceptually similar patches to a single one, and leave unchanged perceptually dissimilar patches.

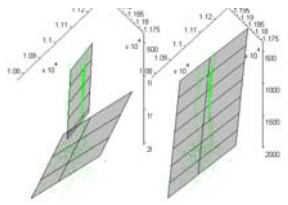

Fig. 6: Merging of two patches (left) to a single one (right). The merged patch results from fitting a single plane to the union of supporting points of the two original patches, followed by trimming.

Merging is derived from 2D line merging algorithms based on principles of perceptual grouping [10]. Intuitively spoken, the underlying similarity measure takes into account the closeness, coplanarity and angle between normals of two patches. Note that the similarity therefore is based on the model, not on the dataset to be fit.
Merging of a single pair s_a , s_b of patches is processed in two steps.

1. determine similarity
2. if sufficiently similar: merge by creating a single patch using least square fitting and trimming

Step 1: similarity determination
The patches have to be sufficiently close. This is defined by checking for overlap of bounding cubes of each patch, expanded by G (grid size) in each direction. If this condition is met, then the patches are tested if they are also sufficiently coplanar and parallel. To determine this, the points $a_i \in S^r(s_a)$ supporting s_a are projected onto the plane e_b containing s_b and vice versa ($b_j \in S^r(s_b)$ are projected onto e_a). The mean distance of the points to their projections is computed for both sets, resulting in two values d_a, d_b. If $min(d_a, d_b) < G$ the patches will be merged. Note that again the parameter G plays an important role.

Step 2: merge
A new plane is fit to the set union $S^r(s_a) \cup S^r(s_b)$ with a classical regression and trimmed to a patch.
Merging is done iteratively until all possible pairs of patches are sufficiently dissimilar.

E. DETERMINATION AND INFLUENCE OF THE GRID SIZE PARAMETER G

A central parameter in the modified SMEMPF process is the grid size G. Apart from its visible manifestation as the edge length of each tile of a patch, it controls different instances in the process:

- the tiling size (as mentioned)
- the distinction between supported/unsupported tiles
- indirectly, by determining tile support, it steers the split process
- it determines candidates for merging

G is newly determined in each iteration of the EMPF process, based on the distribution of the distances of data points to patches/tiles they support. It is computed as $3std(d)$, where $std(d)$ is the standard deviation of point patches distances computed for all data points to the patches they support. G can be seen as quality of approximation of the patches to the data, a lower value shows a better approximation. Note that this measure of quality is only a valid measure if the number of patches is fixed: increasing the number of patches, e.g. to a value such that each patch is exactly supported by 3 data points yields $G=0$, but simply measures overfit to noise instead of data. Experimental results show, that the interplay between split and merge leads to a decreasing value of G in each iteration step in the beginning of the SMEMPF process (when starting with a high value) until it balances itself. The decrease is due to two facts:

1. The re-computation of G is made using the reduced support set. Since this is a subset of the closest points in the support set, the standard deviation is smaller than the standard deviation of the (unreduced) support set.
2. But mainly: split leads to a better fit.

In contrast, the merge process increases the standard deviation due to a less optimal fit. Hence, as long as, intuitively, there's a stronger rate of split than merge, as it naturally is in the first iteration steps if the process is started with a low number of patches, G decreases until it balances itself to an appropriate value due to the increasing force of merge during the iteration.

III. EXPERIMENTAL RESULTS

The final result of the SMEMPF algorithm consists of coplanar point sets, we use orthogonal projection to project the reduced support sets $S^r(s_j)$ onto their supported patches s_j. The result therefore can be seen as a segmentation of the original point set into sets of replaced (projected) data points, related by their support to the same patch. Non assigned data points are seen as outliers and are dropped. Fig. 8 shows SMEMPF using an outdoor dataset, fig. 9 illustrates the final result. We conducted two experiments to show the performance of the algorithm.

Experiment I: angular distance, ground truth dataset
This experiment generates 25 sets $N_{1..25}$ of ~7000 points each by randomly sampling a 3D model of 4 walls with a different amount of Gaussian relocation (replacement noise), see fig. 7. The standard deviation of point distances to the ground truth walls is 1-25 for N_1 to N_{25} respective, i.e. $std(N_i)=i$. Therefore the representation of the walls by the point clouds is more blurred in higher indexed data sets, see fig. 7 C,D,E for examples of N_2, N_{15}, N_{23}. The length of the short edges of the walls forming the small corner (front left in fig.7 a) is 150 units, therefore the structure of this corner gets lost in the point cloud representation with high standard deviation (15-25 units, point sets N_{15} to N_{25}). For each set $N_{i=1..25}$ the EMSMPF is processed, initialized with a set of 3 patches, see fig. 7B. Similar to the evaluation used in [7], the result is analyzed by summing the angular distances of all patch normals to their corresponding ground truth, giving the quality of approximation. Fig. 7F shows a graph of the resulting differences for the 25 sets. It can be seen that the approximation is constantly good even up to a high noise level (std=15, see fig. 7D,F). The value of a difference of ~4 degrees shows an average difference of 1 degree for each patch. The jump in values for N_{22} to N_{23} result from wrong fitting: the high amount of noise destroyed the corner feature, see fig. 7E,F.

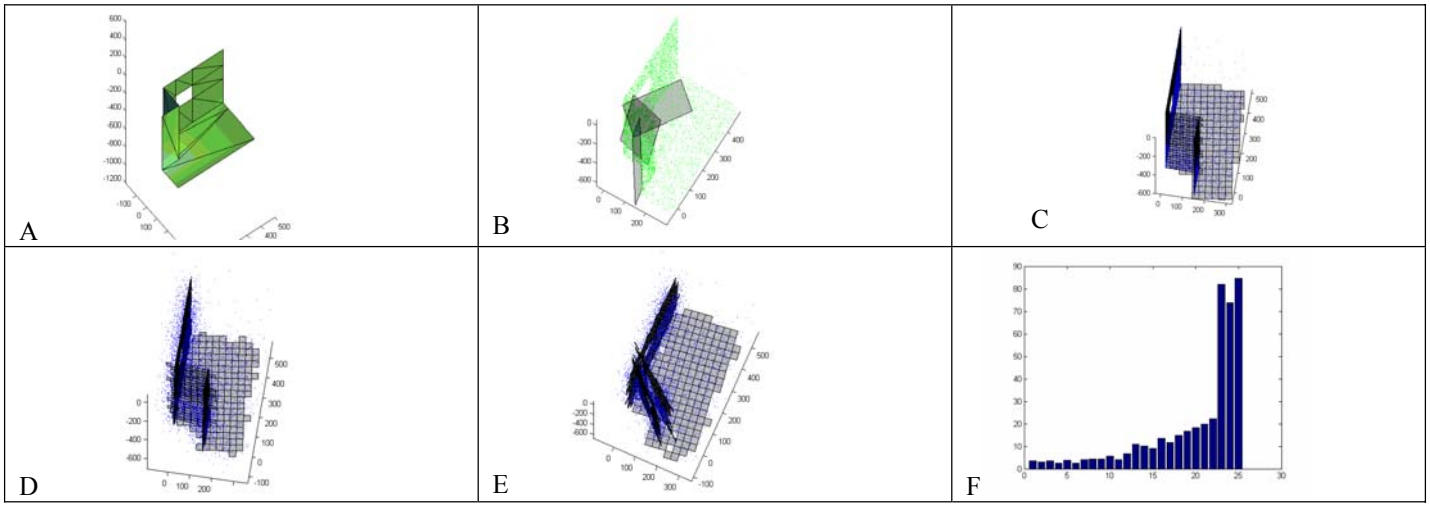

Fig. 7: Fitting to a ground truth dataset. A) the original model (4 walls) B) set (no noise) of simulated scan points created from A) and initial 3 patches. The figures C)-E) show the result of the SMEMPF algorithm (each after 6 iterations), using the initial patch configuration shown in B) but with random replaciement of scan points. C) replacement with standard deviation of 2, D) std = 15, E) std= 23. The length of the short walls being misrepresented in E) is 150, the representation fails due to the high amount of noise, the original structure is not visible in the point set. F) Sum of angular errors of patch normals. Y-axis: angular error, x-axis: standard deviation of replaced data points. The values at x=2, 15, 23 correspond to C), D) E)

Experiment II: outdoor dataset

The dataset, taken by a real mobile robot equipped with a 3D laser scanner on Stanford Campus, consists of ~100000 data points. The points, resulting from multiple scans, are aligned using the technique explained in [5]. Due to the mechanical properties of the robot the dataset is extremely noisy, e.g. points corresponding to the (planar) side wall of the archway (see fig. 8) create a point cloud of width about 50cm. The initialization for the experiment is a set of 10 randomly placed patches. Fig. 8 shows a) initialization, B)-C) patches and supported tiles after iterations 4 and 8 (final iteration). The result contains 61 patches. The direction of the patches' bounding edges is determined by the principal axis of the supporting dataset (see section II.A), and is therefore not always in correspondence with the visual expectation. This has no influence on the final result, shown in fig. 9, consisting of the reduced support sets $S^r(e_j)$ projected onto their corresponding planes e_j. Different colors mark different sets. The final result decomposes the original point set into subsets of points belonging together (i.e. supporting one patch), the points of each subset being relocated to be coplanar. It can be seen that the algorithm found a fit using 61 patches, segmenting the original dataset into planar objects, adjusting the initial under estimation of 10 planes in random locations to an intuitive solution. The (planar) point sets achieved can be processed further, e.g. to find the outlines of objects.; analysis of the corresponding supporting points can detect planar/non planar objects in the data set: the standard deviation of the (non reduced) support set gives immediate information about the planarity, enabling the system to distinguish between walls, trees etc., a topic out of the scope of this paper.

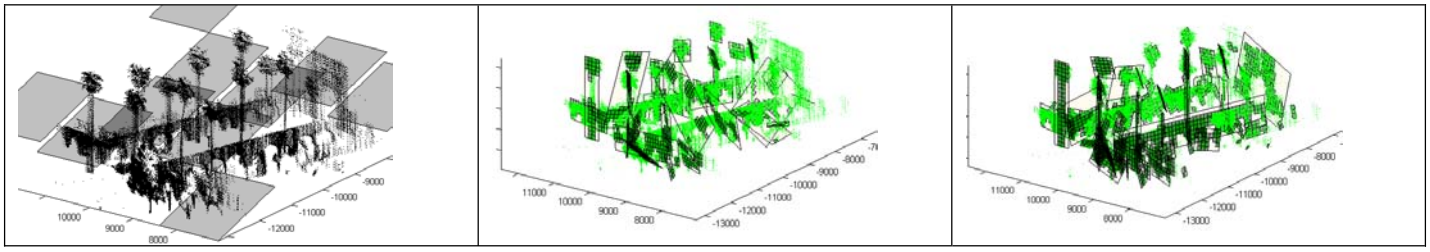

Fig. 8: Performance of SMEMPF on the data set Stanford Campus: A) data set and initial patches (10 patches, randomly selected) B) after iteration 4, C) final result: iteration 8, 61 patches. See also fig. 9 for a different visualization of C).

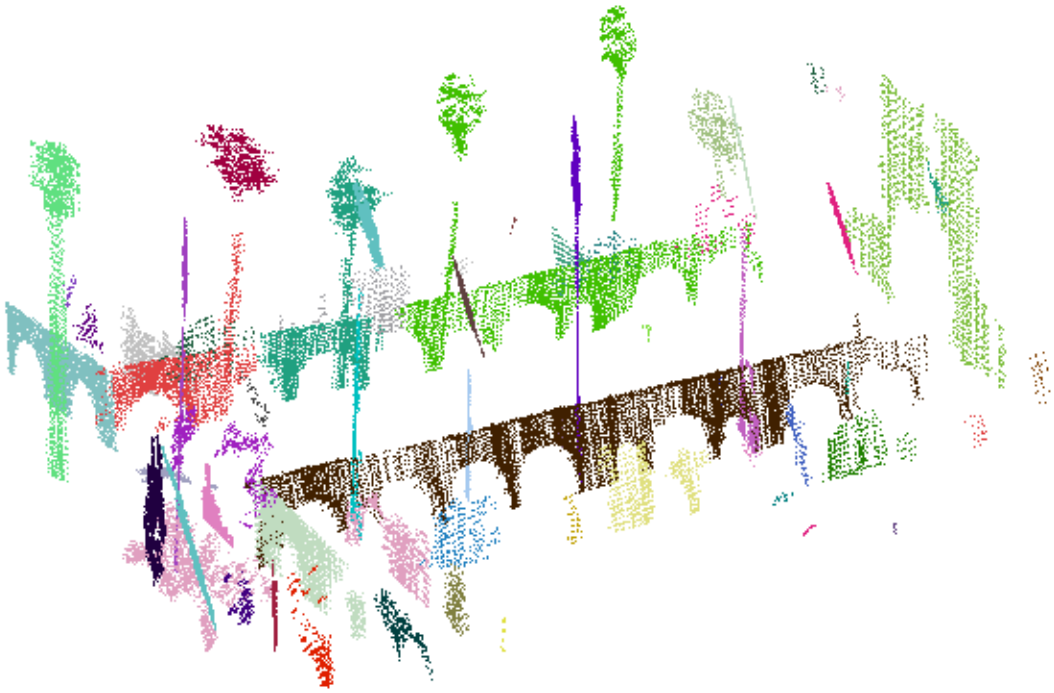

Fig. 9: final result of SMEMPF on Stanford Campus. Shown are the reduced support sets $S^r(e_j)$ projected onto their corresponding planes e_j, using the points and patches of the result shown in fig. 8 C). Different colors mark different sets. The final result decomposes and relocates the original point set into coplanar subsets of points belonging to a single planar structure.

IV. CONCLUSION

The SMEMPF, used as a mapping procedure in robotics as a combination of Expectation Maximization patch fitting with alternating patch splitting and merging was proven to be a powerful tool to gain a patch representation of maps formerly consisting of independent LASER range scanner reflection points. The newly introduced merging step balances the number of patches, created by splitting, in a visually natural way and therefore allows for the number of starting patches for the EM step to be highly imprecise.

V. REFERENCES

1. A. Blake and B. Freeman, "Learning and vision: Generative methods,"in *Lecture at ICCV*, 2003.

2. Dempster, N. Laird, and D. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the Royal Statistical Society, Series B*, vol. 39(1), p. 138, 1997.

3. M. J. R. Healy and M. Wesmacott, "Missing values in experiments analyzed on automatic computers," *Appl. Statist.*, vol. 5, pp. 203-206,1956.

4. S. Thrun, C. Martin, Y. Liu, D. Hähnel, R. Emery-Montemerlo, D. Chakrabarti, and W. Burgard. "A real-time expectation maximization algorithm for acquiring multi-planar maps of indoor environments with mobile robots." *IEEE Transactions on Robotics and Automation*, 2003.

5. S. Thrun, D. Hähnel, D. Ferguson, M. Montemerlo, R. Triebel, W. Burgard, C. Baker, Z. Omohundro, S. Thayer, and W. Whittaker "A system for volumetric robotic mapping of underground mines." In *IEEE International Conference on Robotics and Automation (ICRA)*, Taipei, Taiwan, 2003. ICRA.

6. S. Thrun, W. Burgard, and D. Fox. "A real-time algorithm for mobile robot mapping with applications to multi-robot and 3D mapping." In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, San Francisco, CA, 2000. IEEE.

7. R. Triebel, W. Burgard and F. Dellaert , "Using Hierarchical EM to Extract Planes from 3D Range Scans". In Proc. of the IEEE International Conference on Robotics and Automation (ICRA), 2005

8. M. Veeck and W. Burgard, "Learning polyline maps from range scan data acquired with mobile robots," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2004.

9. Weingarten, J. and Siegwart, R. "EKF-based 3D SLAM for Structured Environment Reconstruction". In *Proceedings of IROS*, Edmonton, Canada, August 2-6, 2005. (IROS'2005)

10. M. Wertheimer, "Maximum likelihood from incomplete data via the EM algorithm," *Psycologische Forschung*, vol. 4, pp. 301–350, 1923.

11. Denis F. Wolf, Andrew Howard, and Gaurav S. Sukhatme, "Towards Geometric 3D Mapping of Outdoor Environments Using Mobile Robots," In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1258-1263, Aug 2005

12. C. Früh and A. Zakhor, "An Automated Method for Large-Scale, Ground-Based City Model Acquisition" in *International Journal of Computer Vision*, Vol. 60, No. 1, October 2004, pp. 5 - 24