Plane-extraction from depth-data using a Gaussian mixture regression model

Richard Marriott, Alexander Pashevich, Radu Horaud

Plane-extraction from depth-data using a Gaussian mixture regression model

Richard T. Marriott[a], Alexander Pashevich[a], Radu Horaud[a]

[a] *INRIA Grenoble Rhône-Alpes & Univ. Grenoble Alpes, France*

Abstract

We propose a novel algorithm for unsupervised extraction of piecewise planar models from depth-data. Among other applications, such models are a good way of enabling autonomous agents (robots, cars, drones, etc.) to effectively perceive their surroundings and to navigate in three dimensions. We propose to do this by fitting the data with a piecewise-linear Gaussian mixture regression model whose components are skewed over planes, making them flat in appearance rather than being ellipsoidal, by embedding an outlier-trimming process that is formally incorporated into the proposed expectation-maximization algorithm, and by selectively fusing contiguous, coplanar components. Part of our motivation is an attempt to estimate more accurate plane-extraction by allowing each model component to make use of all available data through probabilistic clustering. The algorithm is thoroughly evaluated against a standard benchmark and is shown to rank among the best of the existing state-of-the-art methods.

1. Introduction

The objective of this paper is to construct simple planar models of environments by identifying flat surfaces within depth-data. We propose to do this by (i) fitting the data with a piecewise-linear Gaussian mixture regression (GMR) model – a Gaussian mixture model (GMM) whose components are *skewed* over planes, making them flat in appearance rather than being ellipsoidal; and then (ii) selectively *fusing* contiguous, coplanar components. Part of our motivation for evaluating this method was to attempt to estimate more accurate model parameters by allowing each model component to make use of all available data through probabilistic clustering. This contrasts with most other recent methods (Enjarini and Gräser, 2012), (Feng et al., 2014), (Holz and Behnke, 2013), (Holz et al., 2011), (Hulik et al., 2012), (Oehler et al., 2011) which, for the sake of efficiency, compromise by working with noisier subsets of data-points. The application in which we are specifically interested is the perception of a 3D environment by a non-human observer in order to enable navigation within that environment. The observer may be a wheeled or a legged robot, a drone, a driver-less car, a human perception-aid such as that seen in (Pradeep et al., 2013), or any other similar device.

Recently, dense depth-data have become readily available due to the development of affordable structured light and time-of-flight cameras. Each of these sensor-types produces images of depth-related values that can be projected as clouds of 3D points. These point-clouds, however, are nothing more than a noisy set of points that only sample the environment. The observer must then be able to make sense of these observations by using them to construct a model of some form, e.g. a set of planar surfaces.

An alternative to a piecewise-planar model might be to attempt to represent the environment as a set of known objects. To do so, however, comprehensive object-recognition training would be required. In practice, in a dynamic, real-world environment, such a technique would ultimately only be able to complement a more general, unsupervised approach. Planar primitives are sufficiently general to model most environments. They are particularly appropriate in the home and office, where planar surfaces are prevalent, but can also handle more complex scenes, approximating curved surfaces in a piecewise fashion. Although a piecewise planar representation of the environment may not allow many objects to be iden-

tified, it provides a certain set of very useful semantics. Namely, the observer knows that it can navigate safely on roughly horizontal planes and that it cannot pass through roughly vertical ones.

The main contribution of our paper is a probabilistic treatment of the problem of extracting planes from depth images. We propose to combine linear regression with GMM (Deleforge et al., 2015), thus yielding an expectation-maximization (EM) algorithm, with proven mathematical convergence, that deterministically clusters the 3-D data into 2-D Gaussian components via likelihood maximization. Moreover, we use a recently proposed trimming method (Galimzianova et al., 2015) that, unlike random sampling such as RANSAC-based methods, can be embedded within EM in a principled way. We demonstrate, using a standard benchmark, that accuracy of depth-image segmentation by our robust GMR technique is comparable with the best of the other state-of-the-art methods.

2. Related work

There are many different methods of plane-extraction. These methods tend not to rely on single concepts but, instead, combine various *component-algorithms* in different ways. There are three components that are typically used: (i) Pixel-clustering; (ii) Region-growing, whether it be to grow regions pixel by pixel or to absorb some form of nearby superpixels; and (iii) RANSAC plane-fitting, usually applied to local regions only (Enjarini and Gräser, 2012) (Hulik et al., 2012) (Oehler et al., 2011).

In (Feng et al., 2014) and (Holz and Behnke, 2013), various region-growing concepts are used. E.g. (Holz and Behnke, 2013) performs per-pixel region-growing based on per-point normal-orientation and combined mean squared error (MSE). A second, larger-scale merging of regions is then performed to collect together planes that may have become disjoint due to noise in the original surface-normals. In (Feng et al., 2014), some of the noise of per-point normal-estimation is reduced by first creating a grid of superpixels organised in an adjacency graph. Agglomerative hierarchical clustering[1] (AHC) is then used to merge the superpixels followed by per-pixel region-growing to refine the *sawtooth* edges caused by the initial grid.

There are many examples of algorithms that perform clustering. In (Holz et al., 2011), per-pixel normal-estimation is performed and then clustering by discrete values of normal-orientation and of perpendicular distance to the origin. Further pixel-by-pixel refinement is then performed to capture those points falling just on the wrong side of the discretization boundaries from the value of a dominant plane. In (Enjarini and Gräser, 2012) the gradient of depth (GoD) features are clustered: Points belonging to the same plane will have the same GoD across them. Once clusters are found, RANSAC plane-fitting is applied followed by merging of nearby planes. In (Pham et al., 2016) an adjacency graph is constructed over local surface patches and a graph clustering algorithm is then applied. Plane extraction is formulated as the minimization of a global pairwise energy function which jointly considers plane fidelities and geometric consistencies between planes, i.e. orthogonal or parallel planes.

A standard plane-extraction approach is to run RANSAC sequentially until no more planes can be found. (Hulik et al., 2012) and (Oehler et al., 2011) use RANSAC for robust plane-fitting, applying it to local regions only, for efficiency. Clusters belonging to the resulting planar components are then grown to include surrounding points. (Oehler et al., 2011) finds the initial local regions via a Hough transform-based pre-segmentation. In (Gallo et al., 2011) RANSAC is applied to connected components of inliers. In (Qian and Ye, 2014) a coherence check is performed to remove data patches whose normals are in contradiction to the fitted planes, followed by a recursive plane-clustering process.

In this work, we introduce the *robust piecewise-linear* Gaussian mixture regression (RPL-GMR) algorithm for optimally fitting a set of planes to a 3D point cloud. The algorithm contains an outlier-trimming process, thus being able to replace RANSAC. In the literature, there are very few examples of using mixture models for plane-extraction. One example is (Liu et al., 2001). Note, however, that the model used in (Liu et al., 2001) is a mixture of *unbounded* planes that extend throughout the whole

[1] Despite it's name, the AHC in (Feng et al., 2014) is actually performing region-growing on a set of superpixels due to the restriction of the adjacency graph.

data-set. The idea of plane-locality, which is essential for good performance in more complex environments, is only introduced as a post-processing step. The RPL-GMR formulation is such that the locality of planes is estimated simultaneously with the planar parameters, making RPL-GMR a more powerful and elegant alternative to existing methods.

The rest of the paper is organised as follows: Section 3 gives the RPL-GMR formulation and its associated EM algorithm; Section 4 contains details of the various stages of the algorithm; in Section 5 our algorithm is evaluated against various others using the *SegComp* data-set (Hoover et al., 1996); and in Section 6 we draw conclusions.

3. Piecewise-linear Gaussian mixture regression

The proposed model is a form of constrained GMM to find planar patches within sets of 3D data-points. A standard GMM would not be particularly useful and would find ellipsoid-like densities in the data. The model of (Deleforge et al., 2015), on the other hand, makes the assumption that data in high-dimensional space lie on a lower-dimensional manifold (corrupted only by uncorrelated Gaussian noise), and furthermore, that the surface can be well-approximated by a patchwork of locally linear functions. A model that makes these assumptions is ideal in our case where we have data-points measured at the 2D manifold which is the *visible frontier* of the scene, and where we have scenes containing many planes, i.e. locally linear functions in the manifold.

Let this manifold be described by a function $g : \boldsymbol{\mathcal{X}} \mapsto \boldsymbol{\mathcal{Y}}$ where $\boldsymbol{\mathcal{X}} \subset \mathbb{R}^2$ and $\boldsymbol{\mathcal{Y}} \subset \mathbb{R}$. Obviously, $g(\mathcal{X})$ is not necessarily linear, in our case being composed of surfaces with various characteristics. Let $\mathbf{x} \in \mathcal{X}$ and $y \in \mathcal{Y}$ be realisations of the random variables $\mathbf{X} \in \mathbb{R}^2$ and $Y \in \mathbb{R}$. The proposed model approximates the potentially nonlinear $g(\mathbf{x})$ in a piecewise linear fashion. As is common practice in mixture models, a discrete, hidden variable, $Z \in \mathbb{N}$ is introduced. The complete data then become $(\mathbf{X}, Y, Z)$ where a realisation $(\mathbf{x}, y, Z = k)$ of $(\mathbf{X}, Y, Z)$ indicates that y is related to $\mathbf{x}$ by a affine mapping indexed by k, plus some error term, e_k. We assume, then, that $g(\mathbf{x})$ can be approximated by the following mixture of affine transformations:

$$Y = \sum_{k=1}^{K} \mathbb{I}(Z = k)(\mathbf{A_k}\mathbf{X} + b_k + e_k) \tag{1}$$

where $\mathbb{I}$ is an indicator function such that $\mathbb{I}(Z) = 1$ if $Z = k$, or 0 otherwise; $\mathbf{A}_k \in \mathbb{R}^2$ and $b_k \in \mathbb{R}$ are the mapping parameters of the k-th affine transformation; and $e_k \in \mathbb{R}$ is a term capturing errors both in the observations and in the mapping. Let the joint variable $(\mathbf{X}, Y)$ be modeled by a GMM:

$$p(\mathbf{x}, y; \theta) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}, y; \mathbf{m}_k, \mathbf{V}_k) \tag{2}$$

where π_k, $\mathbf{m}_k$ and $\mathbf{V}_k$ are the priors, means and covariances of the mixture, respectively. This is equivalent to:

$$p(\mathbf{x}, y; \theta) = \sum_{k=1}^{K} p(y|\mathbf{x}, Z = k; \theta) p(\mathbf{x}|Z = k; \theta) p(Z = k; \theta) \tag{3}$$

These probability distributions can be modeled as Gaussians, and so we have:

$$p(y|\mathbf{x}, Z = k; \theta) = \mathcal{N}(y; \mathbf{A}_k\mathbf{x} + b_k, \sigma_k) \tag{4}$$

$$p(\mathbf{x}|Z = k; \theta) = \mathcal{N}(\mathbf{x}; \mathbf{c}_k, \mathbf{\Gamma}_k) \tag{5}$$

$$p(Z = k; \theta) = \pi_k \tag{6}$$

where $\mathbf{c}_k \in \mathbb{R}^2$ and $\mathbf{\Gamma} \in \mathbb{R}^{2\times2}$ are, respectively, the centre and covariance of the Gaussian components in the space of $\boldsymbol{\mathcal{X}}$. Combining (3), (4), (5) and (6), we get the explicit expression for the joint probability of the observed data $p(\mathbf{x}, y; \theta) = \sum_{k=1}^{K} \pi_k \mathcal{N}(y; \mathbf{A}_k\mathbf{x} + b_k, \sigma_k)\mathcal{N}(\mathbf{x}; \mathbf{c}_k, \mathbf{\Gamma}_k)$ This is equivalent to the Gaussian distribution of the joint variable $(\mathbf{X}, Y)$ in equation (2) where the mean and covariance are given by

$$\mathbf{m}_k = \begin{bmatrix} \mathbf{c}_k \\ \mathbf{A}_k\mathbf{c}_k + b_k \end{bmatrix}, \mathbf{V}_k = \begin{bmatrix} \mathbf{\Gamma}_k & \mathbf{\Gamma}_k\mathbf{A}_k^T \\ \mathbf{A}_k\mathbf{\Gamma}_k & \sigma_k + \mathbf{A}_k\mathbf{\Gamma}_k\mathbf{A}_k^T \end{bmatrix} \tag{7}$$

The parameter set is $\theta = \{\mathbf{c}_k, \mathbf{\Gamma}_k, \mathbf{A}_k, b_k, \sigma_k, \pi_k\}_{k=1}^{K}$.

The PRL-GMR algorithm is an EM procedure that iteratively maximises the expectation of the complete-data log-likelihood with respect to the probability distribution

of the hidden variables given the current model parameters:

$$\mathcal{L}(\theta) = \sum_{k=1}^{K} \frac{1}{r_k} \sum_{n=1}^{N} r_{nk} \log(p(\mathbf{x}_n, y_n, Z_n = k; \theta)) \quad (8)$$

where $r_k = \sum_{n=1}^{N} r_{nk}$ and r_{nk} are the *responsibilities*:

$$r_{nk} = \frac{\pi_k \mathcal{N}(y_n; \mathbf{A}_k \mathbf{x}_n + b_k, \sigma_k) \mathcal{N}(\mathbf{x}_n; \mathbf{c}_k, \mathbf{\Gamma}_k)}{\sum_{i=1}^{K} \pi_i \mathcal{N}(y_n; \mathbf{A}_i \mathbf{x}_n + b_i, \sigma_i) \mathcal{N}(\mathbf{x}_n; \mathbf{c}_i, \mathbf{\Gamma}_i)} \quad (9)$$

Maximizing (8) with respect to each of the model parameters in θ we obtain the parameter-update equations below:

$$\mathbf{c}_k = \sum_{n=1}^{N} \frac{r_{kn}}{r_k} \mathbf{x}_n, \quad (10)$$

$$\mathbf{\Gamma}_k = \sum_{n=1}^{N} \frac{r_{kn}}{r_k} (\mathbf{x}_n - \mathbf{c}_k)(\mathbf{x}_n - \mathbf{c}_k)^T \quad (11)$$

$$\mathbf{A}_k = \overline{Y}_k \overline{\mathbf{X}}_k^{\dagger}, \quad (12)$$

$$\mathbf{b}_k = \sum_{n=1}^{N} \frac{r_{kn}}{r_k} (y_n - \mathbf{A}_k \mathbf{x}_n) \quad (13)$$

$$\sigma_k^2 = \sum_{n=1}^{N} \frac{r_{kn}}{r_k} (y_n - \mathbf{A}_k \mathbf{x}_n - b_k)^2 \quad (14)$$

$$\pi_k = r_k / \sum_{k=1}^{K} r_k \quad (15)$$

where $\dagger$ is the Moore-Penrose pseudo inverse operator and $\overline{\mathbf{X}}_k = \{\sqrt{r_{nk}}(\mathbf{x}_n - \mathbf{c}_k)\}_{n=1}^{N}$, $\overline{Y}_k = \{\sqrt{r_{nk}}(y_n - \bar{y}_k)\}_{n=1}^{N}$ are sets of centred and weighted points with $\bar{y}_k = \sum_{n=1}^{N} \frac{r_{nk}}{r_k} y_n$. The RPL-GMR algorithm should be evaluated until convergence of the expected complete-data log-likelihood in (8). A typical convergence criterion might be

$$\mathcal{L}(\theta^{(i)}) - \mathcal{L}(\theta^{(i-1)}) < \epsilon \left| \mathcal{L}(\theta^{(i-1)}) \right| \quad (16)$$

where (i) denotes the iteration index and ϵ is some constant to be specified.

4. Implementation details

We now describe in detail the implementation of the proposed method. A formal description is provided in Algorithm 1 and the effect of each of the stages can be seen in Fig. 1.

4.1. Initialisation

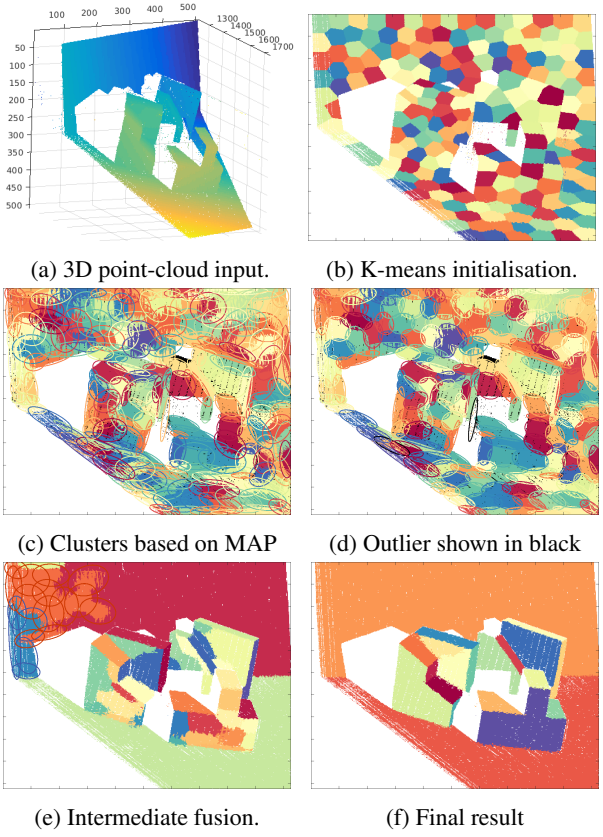
The RPL-GMR algorithm (as with any EM algorithm) does not necessarily find globally optimal solutions and is therefore sensitive to initial conditions. An important aspect of initialisation is the decision of how big a model to use in terms of the number of components. A sure way to reach an undesirable solution would be to initialise the algorithm with too few model components. We clearly need to fit a model with at least as many components as there are significant planar surfaces in the depth image. However, as this number is not known *a priori*, we choose a large number of model components that is likely to be higher than the number of planes we expect to find, relying on our fusing procedure to later reduce the number of components where necessary.

Plane-size is also an important consideration when deciding on the number of model components for the following reason: Whereas errors in the positions of points *across* planes may obey something like Gaussian distributions, the positions of points *along* planes have distributions that are more uniform in nature. Non-Gaussian distributions can be better described by multiple Gaussians. As a result, our Gaussian components often *prefer* to colocate, sharing points belonging to a single plane, rather than forcing each other to occupy different planes. With components not always readily re-distributing to other regions, it is important that components are placed with good proximity to all planes during initialisation. For this reason, if a model with too few components is used, datapoints belonging to smaller planes will often be neglected. Choosing a relatively large number of initial model components is one way to ensure that smaller planes are also captured. On the other hand, fitting too many model components is computationally expensive and can lead to over-fitting where components fit to noise, ignoring larger-scale patterns in the data. The choice of the number of model components is therefore data-dependent and is a hyper-parameter that must be tuned.

Initial model parameters are calculated from clusters found by applying randomly initialised k-means to the 3D point set. An example of output of this initialisation procedure is shown in Fig. 1b. Also tested was initialisation using points within squares of a regular grid. RPL-GMR was found to converge more quickly when initialised with k-means than with the regular grid; perhaps because, despite not *knowing* about planes in the data, k-means is still

Algorithm 1 RPL-GMR

procedure RPL-GMR(X, Y, k_{max}, ϵ)
  $[\pi, c, \Gamma, A, b, \sigma] = KMeansInit(X, Y, k_{max})$
  $L(\theta^{t-1}) = 0$
  ▷ INITIAL EXPECTATION STEP
  $u_{nk} = p(y_n|x_n, Z = k; \theta)p(x_n|Z = k; \theta)$
  $r_{nk} = \pi_k u_{nk}$
  $r_n = \sum_{k=1}^{K} r_{nk}$
  repeat
    $r_{nk} = r_{nk}/r_n$ ▷ Normalise
    ▷ TRIMMING STEP
    $(X', Y') = trimming(X, Y, r_{nk}, u_{nk}, L(\theta^{t-1}))$
    $r_k = \sum_{n'=1}^{N'} r_{n'k}$
    $\pi_k = r_k / \sum_{k=1}^{K} r_k$
    ▷ MAXIMISATION STEP (using [X', Y'])
    Evaluate equations (10)-(15) to improve θ
    ▷ EXPECTATION STEP (using [X, Y])
    $u_{nk} = p(y_n|x_n, Z = k; \theta)p(x_n|Z = k; \theta)$
    $r_{nk} = \pi_k u_{nk}$ ▷ Don't normalise yet
    $r_n = \sum_{k=1}^{K} r_{nk}$
    $L(\theta^t) = \sum_{n=1}^{N} \log(r_n)$
  until $L(\theta^t) - L(\theta^{t-1}) < \epsilon |L(\theta^{t-1})|$
  ▷ POST-PROCESSING
  $r_{nk} = r_{nk}/r_n$ ▷ Normalise
  $r_k = \sum_{n'=1}^{N'} r_{n'k}$
  $\pi_k = r_k / \sum_{k=1}^{K} r_k$
  $clustering_{n'} = max_k(r_{n'k})$
  $\pi_k = densityCheck(X', clustering_{n'}, \pi_k)$
  $(\theta, X_{seg}) = FuseComponents(\theta, r_k, X, Y)$
return (θ, X_{seg})



(a) 3D point-cloud input. (b) K-means initialisation.
(c) Clusters based on MAP (d) Outlier shown in black
(e) Intermediate fusion. (f) Final result

Figure 1: Visualisation of various stages of Algorithm 1.

able to capture edges where one plane occludes another and there is a large difference in the proximity of points between planes.

4.2. Robustness to outliers

Data-sets containing outliers can introduce biases into model parameters during plane-fitting and can even lead to completely spurious planes being found. As mentioned in Section 2, many plane-extraction methods achieve robustness by fitting planes using RANSAC. Instead, we embed a *trimming* step, inspired by (Galimzianova et al., 2015), within the body of the EM procedure: At each iteration of RPL-GMR, points are ranked based on how likely they are to be outliers, then a certain fraction are discarded (or *trimmed*) from the bottom of the ranking before continuing to the maximisation step. The general assumption made during trimming is that the parameters of the model are initialised (and remain) to be close to their ideal values. If this is true then outliers can be identified based on their agreement (or lack of agreement) with the current estimate of the model.

To trim successfully, we need two things: (i) reasonable knowledge of the number of outlying data-points; and (ii) a score by which the data-points can be ranked in order of likelihood that they are outliers. Knowledge of the number of outlying data-points can be obtained from training data or else from known camera-characteristics. It is better to over-estimate this fraction (Galimzianova et al., 2015). As for the score, (Galimzianova et al., 2015) rec-

ommends that, for unbalanced Gaussian mixtures – i.e. mixtures where components represent different numbers of data-points - component-wise confidence-level ordering based on Mahalanobis distance from the most likely component-centre should be used. This avoids the trimming of all points belonging to weak components, as might occur with ordering based on posterior probabilities. Posterior probabilities *are* used, however, to associate points with most likely components. Rather than ordering based on Mahalanobis distances, we use the likelihood, $p(\mathbf{x}, y|Z = k; \theta)$, since it is already calculated prior to the trimming step (u_{nk} in Algorithm 1). This ordering is equivalent to ordering by Mahalanobis distances since, for Gaussians, Mahalanobis distance is proportional to $\sqrt{-\log p(\mathbf{x}, y|Z = k; \theta)}$, and $\sqrt{-\log(x)}$ decreases monotonically between 0 and 1.

EM guarantees to increase (8) at each iteration. However, by including the trimming step, the data-set used during the maximisation step will likely change at each iteration. This breaks the guarantee of an ever-increasing log-likelihood. To avoid this problem, such that the log-likelihood function can still be used to test for convergence, after having removed a fraction $(1 - \alpha)$ of points, individual points are removed from the sum until the log-likelihood is larger than that following the previous iteration. The maximisation step then improves the parameters to further increase the log-likelihood.

An example of output from RPL-GMR is given in Fig. 1c. For visualisation purposes, clusterings of points based on MAP are represented by different colours. In the same colours, we have also plotted contours of constant probability for each of the **X**-space Gaussians given by equation (5). (The contours each have radii of $c_{D_M} = 2.1$ Mahalanobis distances.) Points that have been trimmed are shown in black, including those of a plane for which a component was unfortunately not found.

4.3. Fusing of planar Gaussian components

At first glance, rather than combining components as a post-processing stage, it might seem that it would be more elegant to include some form of model-selection within the RPL-GMR loop. In (Figueiredo and Jain, 2002), for example, the number of model components is gradually reduced during the EM procedure until the most parsimonious description of the data is found, as measured by a Minimum Message Length (MML) criterion. In our case, however, reducing the number of components based on MML is not meaningful since our distributions of points are non-Gaussian. E.g. many of the planar surfaces are rectangular and are more effectively modelled by multiple components. Rather than reducing the number of components *during* EM iterations, our approach is, instead, to fuse components together as a post-processing stage. By doing so, we are able to obtain more accurate estimates of the plane parameters by combining information from multiple co-planar components, but are also able to maintain the associations of data points with the original set of model components since no further expectation steps are performed following the fusing stage.

Components are fused together if three criteria are met: 1) The components must be adjacent to one another; 2) By combining the components, the RMS probability-weighted deviation of points perpendicular to the combined plane must not exceed a certain threshold; and 3) Each of the components being fused must not protrude too far from the plane of the other component. Similar to (Feng et al., 2014), we first build an adjacency graph of clusters in the 2D **X**-space. In (Feng et al., 2014) this is straightforward as their data are divided into a regular grid. In our case, model components are scattered throughout the data and we must explicitly test for adjacency. To do this, we test for overlap of ellipses formed from the Mahalanobis distances of the Gaussians in (5), scaled by a factor c_{D_M}. An efficient method for testing the overlap of ellipses can be found in (Etayo et al., 2006). An alternative approach could be to test for adjacency of convex cells in the 3D Voronoi tessellation formed by MAP partitioning of the space about the mixture model.

In order to test the second and third fusing-criteria, we make use of the principal components of variation in the data as weighted by the responsibilities found for each component. I.e. for each component, we calculate eigenvalues of the responsibility-weighted data: the smallest eigenvalue is equivalent to the mean squared error (MSE) of points from the plane. In practice we calculate the eigenvalues of matrix (7).

The fusing algorithm proceeds as follows: The node in the adjacency graph whose component has the smallest MSE is identified; *hypothetical* combinations are then made with each adjacent component to find the plane with the lowest combined MSE. No combination is made if the best resulting MSE is greater than a certain threshold,

Method	Correctly detected	Orientation deviation	Over-seg.	Under-seg.	Missed	Spurious
SegComp ABW data-set (30 test images) (Hoover et al., 1996). Scores calculated using a threshold of 80% pixel-overlap.						
USF (Gotardo et al., 2003)	12.7 / 15.2 (83.5%)	1.6	***0.2***	***0.1***	2.1	1.2
WSU (Gotardo et al., 2003)	9.7 / 15.2 (63.8%)	1.6	0.5	0.2	4.5	2.2
UB (Gotardo et al., 2003)	12.8 / 15.2 (84.2%)	**1.3**	0.5	***0.1***	1.7	2.1
UE (Gotardo et al., 2003)	**13.4 / 15.2 (88.1%)**	1.6	0.4	0.2	***1.1***	***0.8***
UFPR (Gotardo et al., 2003)	13.0 / 15.2 (85.5%)	1.5	0.5	0.1	1.6	1.4
Oehler et al. (Oehler et al., 2011)	11.1 / 15.2 (73.0%)	***1.4***	***0.2***	0.7	2.2	***0.8***
Holz et al. (Holz and Behnke, 2013)	12.2 / 15.2 (80.1%)	1.9	1.8	***0.1***	**0.9**	1.3
Feng et al. (Feng et al., 2014)	12.8 / 15.2 (84.2%)	1.7	**0.1**	**0.0**	2.4	**0.7**
RPL-GMR	***13.1 / 15.2 (85.8%)***	1.6	***0.2***	***0.1***	1.8	***0.8***
SegComp PERCEPTRON data-set (30 test images) (Hoover et al., 1996). Scores calculated using a threshold of 80% pixel-overlap.						
USF (Gotardo et al., 2003)	8.9 / 14.6 (60.9%)	2.7	0.4	**0.0**	5.3	3.6
WSU (Gotardo et al., 2003)	5.9 / 14.6 (40.4%)	3.3	0.5	0.6	6.7	4.8
UB (Gotardo et al., 2003)	9.6 / 14.6 (65.7%)	3.1	0.6	***0.1***	4.2	2.8
UE (Gotardo et al., 2003)	10.0 / 14.6 (68.4%)	2.6	**0.2**	0.3	3.8	2.1
UFPR (Gotardo et al., 2003)	**11.0 / 14.6 (75.3%)**	***2.5***	***0.3***	***0.1***	3.0	2.5
Oehler et al. (Oehler et al., 2011)	7.4 / 14.6 (50.1%)	5.2	***0.3***	0.4	6.2	3.9
Holz et al. (Holz and Behnke, 2013)	**11.0 / 14.6 (75.3%)**	2.6	0.4	0.2	**2.7**	**0.3**
Feng et al. (Feng et al., 2014)	8.9 / 14.6 (60.9%)	**2.4**	**0.2**	0.2	5.1	2.1
RPL-GMR	***10.6 / 14.6 (72.4%)***	***2.5***	***0.3***	0.3	***3.0***	***2.0***

Table 1: SegComp benchmarking results using the test data of the ABW and PERCEPTRON datasets. The best results are shown in **bold** and the second-best results are shown in ***slanted bold***. Our method yields very good results (second best and third best in terms of number of correctly detected planes. Overall, RPL-GMR is the second best performing method.

T_{MSE}, or if the third fusing-criterion is not met (discussed below). Fusing will terminate once each combination of adjacent nodes has been tested. In (Feng et al., 2014), the MSEs are stored in a min-heap data-structure for efficiency. This could be done here as well. However, the cost of running our fusing algorithm is already much less than running RPL-GMR.

Up to this point in the algorithm, we have made efforts to ensure that smaller planes in our unbalanced mixture are not lost. For example, one reason we initialise with a large number of components is to capture smaller planes. We also used component-wise confidence-level ordering during trimming to avoid the loss of smaller planes. Without the third fusing-criterion, however, smaller planes could easily be subsumed by larger ones, provided the MSE remains low enough. In some cases the data-based distance metric of combined MSE works well. E.g. dominant planes are able to *mop up* small erroneous planar components fitted to noise at the edges of true clusters, despite having orientations roughly perpendicular to the main plane. If the smaller plane extends significantly beyond the noise of the more dominant plane, however, then we probably don't want to merge the two. Before merging any two components, therefore, we perform the third check on the magnitudes of projections of the two main eigenvectors (in both negative and positive directions) onto the other plane's normal. The test fails if, for both planes, the magnitude of any of these four projections is greater than a certain threshold: $T_{proj} \times \sqrt{\text{MSE}}$.

5. Benchmarking

We evaluated the RPL-GMR algorithm using the *ABW* and *PERCEPTRON* data-sets available as part of the "SegComp" (Segmentation Comparison) project from the University of South Florida (Hoover et al., 1996). Both of these data-sets contain depth-images of entirely planar scenes along with ground-truth segmentations. The images of the ABW data-set were taken using an ABW structured light camera whereas the PERCEPTRON camera uses scanning laser range finding (LRF) technology. Each set contains 10 training-images and 30 test-

images. The SegComp package also includes an automated comparison program that compares segmented images with the ground-truth segmentations and produces various statistics. As well as comparing clustered pixels in the image, the program compares the orientations of the model planes that were found.

When working with depth-images, it is advantageous to be able to use image-coordinates as values in the $\mathbf{X}$-space of the GLLiM model. Doing so avoids potential problems with the degeneracy of points that happen to share the same xy-coordinates in Cartesian space. (In an image, each point has its own, non-degenerate uv-coordinate.) The transformation from image-space to depths, however, is nonlinear. To avoid this problem, it is necessary to work with a quantity that is inversely proportional to depth, such as disparity. In our evaluation we worked with the quantity s/Z using the scale-factor $s = \frac{(\text{rows}+\text{cols})/2}{(1/z)_{max}-(1/z)_{min}}$. Without the scale-factor, inverse depths (which tend to be very small values) have little effect during the EM procedure and the algorithm struggles to differentiate between nearby planes of different orientations. U, V and s/Z are the axes plotted in Fig. 1a.

The parameters involved in our algorithm were tuned by experimenting on the two training sets. The following values were eventually used to process the test data-sets: for ABW $K = 200$, $\epsilon = 10^{-5}$, $MaxIter = 50$, $c_{D_M} = 2.1$, $T_\rho = 0.5$, $T_{proj} = 10$, $T_{MSE} = 5$, $\alpha = 0.98$, and for PERCEPTRON with $T_{MSE} = 7.5$ and $\alpha = 0.99$. The automated results of running on the SegComp test sets are given in Table 1 and a selection of images for direct comparison with those shown in (Feng et al., 2014) are shown in Fig. 2. In all tests, the maximum of 50 RPL-GMR iterations were performed.

Comparison of results in Table 1 shows that RPL-GMR performs consistently well by most of the measures. Out of the nine methods, for the ABW data-set, RPL-GMR ranks second (or joint-second) for four out of the six measures (correct detections, over-segmentation, under-segmentation, and for not producing spurious planes). For the orientation-deviation and missed planes metrics, RPL-GMR ranks lower: joint-fourth and fifth, respectively. However, scores for these metrics are well within the normal range. For the PERCEPTRON data-set, RPL-GMR ranks as second for not producing spurious planes, joint-second for both orientation-deviation and for not missing planes, and third and joint-third for correctly detecting planes and not over-segmenting them. RPL-GMR ranked as only joint-sixth for under-segmentation. However, again, this score is well within the normal range.

For qualitative evaluation, a selection of segmented images is displayed in Fig. 2. In Fig. 2e a large section of a plane has been missed. This seems to have been caused by a combination of unfortunate initialisation and a value of α that was perhaps slightly too small for the image. One solution might be to initialise with a larger number of components, but at greater computational cost. Another problem that can be seen is the under-segmentation of planes in Fig.s 2c, 2g and 2i. These issues seem to have been misdiagnosed by the automated SegComp camparison program as spurious planes since the largest parts of the planes were captured correctly. These problems of over-segmentation could potentially be solved by better tuning of the T_{proj} and T_{MSE} parameters. A coarse hyper-parameter search *was* performed, however.

6. Conclusions

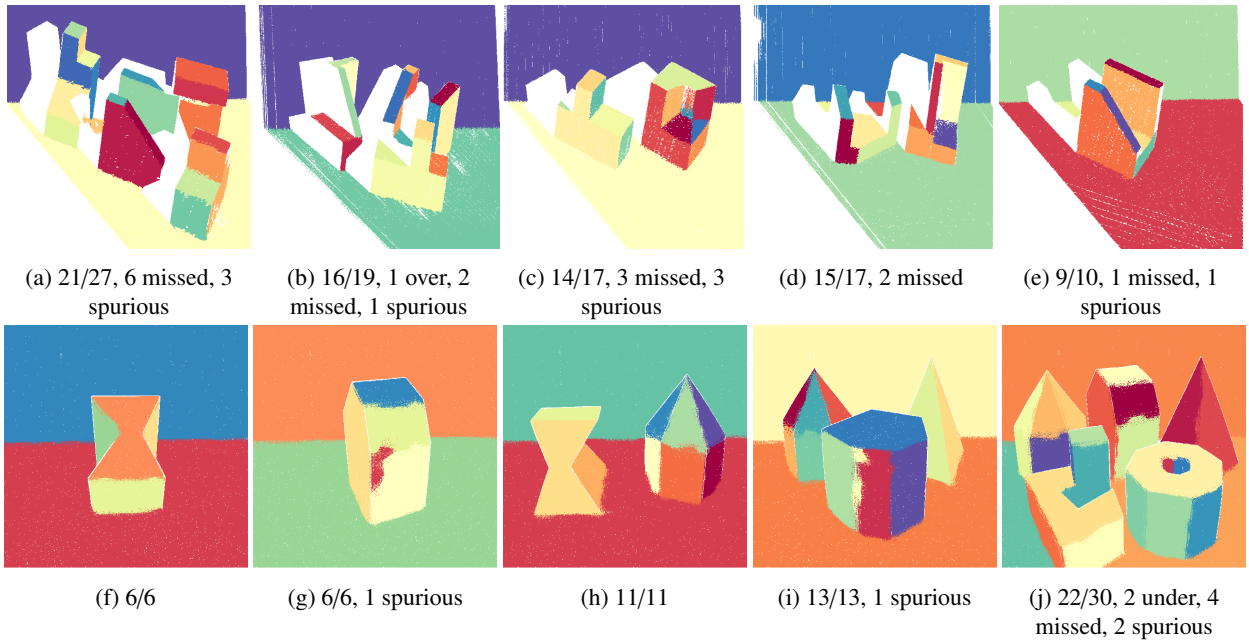
We have shown that the RPL-GMR algorithm can be used successfully to extract planar patches from depth-data. Combined with an outlier-trimming step embedded within the EM procedure to achieve robustness and with the component-fusing method, benchmark results place our algorithm among the top-performing algorithms in the recent literature in terms of segmentation-quality. Compared with other recent methods, RPL-GMR is time-consuming due to the *batch* nature of EM. However, once an initial segmentation has been found, RPL-GMR can be applied incrementally (Evangelidis and Horaud, 2017), thus drastically reducing the computational burden.

Acknowledgments

Financial support from the European Union via the ERC Advanced Grant #340113 (VHIA) is greatly acknowledged.

References

Deleforge, A., Forbes, F., Horaud, R., 2015. High-dimensional regression with gaussian mixtures and partially-latent response variables. Statistics and Computing 25, 893–911.



(a) 21/27, 6 missed, 3 spurious

(b) 16/19, 1 over, 2 missed, 1 spurious

(c) 14/17, 3 missed, 3 spurious

(d) 15/17, 2 missed

(e) 9/10, 1 missed, 1 spurious

(f) 6/6

(g) 6/6, 1 spurious

(h) 11/11

(i) 13/13, 1 spurious

(j) 22/30, 2 under, 4 missed, 2 spurious

Figure 2: Results from the SegComp benchmark using RPL-GMR. Examples from ABW test data (top row) and from the PERCEPTRON test data (bottom row).

Enjarini, B., Gräser, A., 2012. Planar segmentation from depth images using gradient of depth feature, in: IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE. pp. 4668–4674.

Etayo, F., Gonzalez-Vega, L., del Rio, N., 2006. A new approach to characterizing the relative position of two ellipses depending on one parameter. Computer aided geometric design 23, 324–350.

Evangelidis, G.D., Horaud, R., 2017. Joint alignment of multiple point sets with batch and incremental expectation-maximization. IEEE Transactions on Pattern Analysis and Machine Intelligence .

Feng, C., Taguchi, Y., Kamat, V.R., 2014. Fast plane extraction in organized point clouds using agglomerative hierarchical clustering, in: IEEE International Conference on Robotics and Automation, IEEE. pp. 6218–6225.

Figueiredo, M.A.T., Jain, A.K., 2002. Unsupervised learning of finite mixture models. IEEE Transactions on Pattern Analysis and Machine Intelligence 24, 381–396.

Galimzianova, A., Pernuš, F., Likar, B., Špiclin, Ž., 2015. Robust estimation of unbalanced mixture models on samples with outliers. IEEE Transactions on Pattern Analysis and Machine Intelligence 37, 2273–2285.

Gallo, O., Manduchi, R., Rafii, A., 2011. CC-RANSAC: Fitting planes in the presence of multiple surfaces in range data. Pattern Recognition Letters 32, 403–410.

Gotardo, P.F., Bellon, O.R.P., Silva, L., 2003. Range image segmentation by surface extraction using an improved robust estimator, in: IEEE Conference on Computer Vision and Pattern Recognition, IEEE.

Holz, D., Behnke, S., 2013. Fast range image segmentation and smoothing using approximate surface reconstruction and region growing. Intelligent Autonomous Systems , 61–73.

Holz, D., Holzer, S., Rusu, R.B., Behnke, S., 2011.

Real-time plane segmentation using rgb-d cameras, in: Robot Soccer World Cup, Springer. pp. 306–317.

Hoover, A., Jean-Baptiste, G., Jiang, X., Flynn, P.J., Bunke, H., Goldgof, D.B., Bowyer, K., Eggert, D.W., Fitzgibbon, A., Fisher, R.B., 1996. An experimental comparison of range image segmentation algorithms. IEEE Transactions on Pattern Analysis and Machine Intelligence 18, 673–689.

Hulik, R., Beran, V., Spanel, M., Krsek, P., Smrz, P., 2012. Fast and accurate plane segmentation in depth maps for indoor scenes, in: IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE. pp. 1665–1670.

Liu, Y., Emery, R., Chakrabarti, D., Burgard, W., Thrun, S., 2001. Using EM to learn 3D models of indoor environments with mobile robots, in: International Conference on Machine Learning, pp. 329–336.

Oehler, B., Stueckler, J., Welle, J., Schulz, D., Behnke, S., 2011. Efficient multi-resolution plane segmentation of 3D point clouds. Intelligent Robotics and Applications , 145–156.

Pham, T.T., Eich, M., Reid, I., Wyeth, G., 2016. Geometrically consistent plane extraction for dense indoor 3D maps segmentation, in: IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE. pp. 4199–4204.

Pradeep, V., Rhemann, C., Izadi, S., Zach, C., Bleyer, M., Bathiche, S., 2013. MonoFusion: Real-time 3D reconstruction of small scenes with a single web camera, in: IEEE International Symposium on Mixed and Augmented Reality, IEEE. pp. 83–88.

Qian, X., Ye, C., 2014. NCC-RANSAC: A fast plane extraction method for 3-D range data segmentation. IEEE Transactions on Cybernetics 44, 2771–2783.