A method to segment maps from different modalities using free space layout
MAORIS: <u>m</u>ap <u>of</u> <u>r</u>ipples segmentation

Malcolm Mielle, Martin Magnusson, Achim J. Lilienthal

Abstract— How to divide floor plans or navigation maps into semantic representations, such as rooms and corridors, is an important research question in fields such as human-robot interaction, place categorization, or semantic mapping. While most works focus on segmenting robot built maps, those are not the only types of map a robot, or its user, can use. We present a method for segmenting maps from different modalities, focusing on robot built maps and hand-drawn sketch maps, and show better results than state of the art for both types.

Our method segments the map by doing a convolution between the distance image of the map and a circular kernel, and grouping pixels of the same value. Segmentation is done by detecting ripple-like patterns where pixel values vary quickly, and merging neighboring regions with similar values.

We identify a flaw in the segmentation evaluation metric used in recent works and propose a metric based on Matthews correlation coefficient (MCC). We compare our results to ground-truth segmentations of maps from a publicly available dataset, on which we obtain a better MCC than the state of the art with 0.98 compared to 0.65 for a recent Voronoi-based segmentation method and 0.70 for the DuDe segmentation method. We also provide a dataset of sketches of an indoor environment, with two possible sets of ground truth segmentations, on which our method obtains an MCC of 0.56 against 0.28 for the Voronoi-based segmentation method and 0.30 for DuDe.

I. Introduction

State-of-the-art SLAM algorithms enable robots to build metric maps representing their environment. However, robot built maps (referred to as robot maps) are not always easy to understand. They suffer from sensor noise, clutter, and can be complex if the environment is large and complicated. It is often easier for a human to use high-level features, such as rooms and corridors, than to directly be presented with the robot map. This is true in fields such as human-robot interactions, where one could communicate places and directions using rooms' names instead of coordinates, or in planning, where robots need to visit rooms in the optimal order.

But robot maps are not the only type of map a robot can use. For example, hand-drawn sketches are very intuitive interfaces for human-robot interactions and are effective at conveying spacial configurations, as shown by Skubic et al. [1]. However, sketch maps are not metrically accurate, sometimes by design. For example, the person who drew might have altogether ignored a feature of the environment because they judged it unimportant. While humans are able
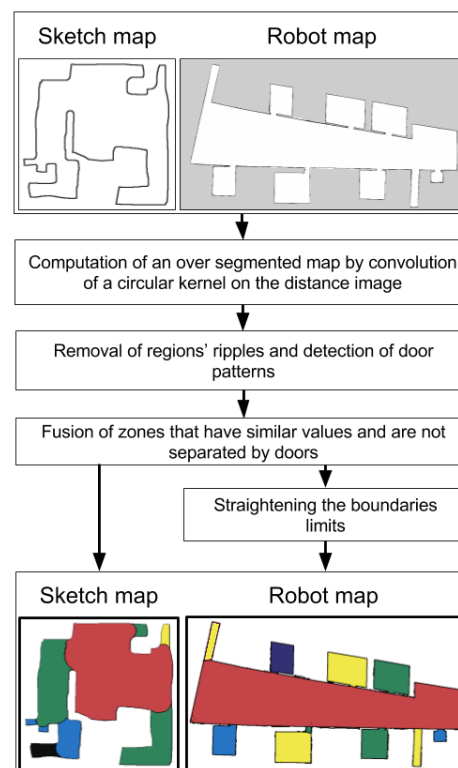

Fig. 1: MAORIS map segmentation process illustrated for a sketch map and a robot map of 2 different environments.

to take those changes into account, and understand the abstraction in the drawing, a robot can not easily interpret such a fuzzy representation of an environment.

In our work, we present a novel method for segmenting maps, even when coming from very different modalities, and, as an application, we focus on robot and sketch maps. An outline of the method is shown in Fig. 1. The algorithm was evaluated on datasets representing both robot maps and sketch maps, and compared to ground truth segmentations given by users. The contributions of this paper are:

- A novel method to extract regions from maps from different modalities. Our method is more robust than the state of the art.
- A dataset of sketches representing three indoor places with human labeled segmentations.
- A discussion on how to evaluate and compare the results of map segmentation algorithms, and a proposed new metric for doing so.

Center of Applied Autonomous Sensor Systems (AASS), Örebro University, Sweden. `firstname.lastname@oru.se`

II. Related Work

Bormann et al. [2] review the literature on room segmentation, and select and implement four algorithms as ROS packages. They provide and compare those methods against 20 floor plans; in most cases a segmentation based on the Voronoi diagram gives the best approximation of the ground truth segmentation. However, the Voronoi based segmentation tends to over-segment the regions, especially corridors. Furthermore, the noisy nature of sketch maps makes it hard to extract the major skeleton of the Voronoi, resulting in over or under-segmentation.

Fermin-Leon et al. [3] propose the DuDe segmentation method that uses a contour based part segmentation [4] for the construction of 2D topological maps. They develop a batch and an incremental algorithm for segmentation and tested them against the ones presented by Bormann et al. [2]. While Voronoi-based segmentation performs better, their method is faster. However, their method over-segments corridors. Also, it is based on contour shape and, when used on sketches with less attention to details, it leads to poor segmentation results as shown in Section IV-C.

Fabrizi et al. [5] use image processing to extract features from an occupancy grid constructed from a sonar. They use fuzzy opening and closure to compute information about the free space and use a watershed algorithm to create regions. Regions are classified into rooms and corridors according to their eccentricity. The watershed uses door patterns to avoid fusing different rooms together. However, those door patterns are not always present in maps.

Park et al. [6] extract maximal empty rectangles in maps and use them for map matching. Since orientation, accuracy, and scale of the maps are unknowns, the final merged map is estimated by minimizing the differences of the factors. This method is dependent on the presence of orthogonal angles in the map, which is not realistic for sketch maps.

Ahmed et al. [7] and Heras et al. [8] discuss a system that uses architectural floor plans with symbols and textual annotations. Their method is made for floor plans with high accuracy and a lot of details, e.g. different types of walls and labels, and is not applicable for sketch and robot maps.

Diosi et al. [9] implement a semi-autonomous room segmentation algorithm. A robot maps an environment while following a user who gives it labels for different locations. Once the mapping is completed, the distance image is used to generate local maxima and they use gradient ascent to group all pixels that move to the same local maxima into regions. The unlabeled segments are merged into the labelled segment that minimizes the distance from the centroid of the unlabelled segment to the closest label. Since this method depends on previous labeling by a user and the position of labels on the map, it is thus not adaptable for sketches and general use cases.

III. Map segmentation

We hypothesize that regions in indoor maps can be found by looking at the layout of free space; more specifically, by looking at changes of sizes between rooms, e.g doors

Data: *map*
Result: *segmented map*
1 Calculate distance image from map;
2 Calculate free space image;
3 Group adjacent pixels of same value in regions;
4 Remove ripples;
5 Merge regions with similar values and not separated a door;
6 Remove regions created by thick walls;
7 **if** *map is a robot map* **then**
8 | Straighten boundaries;
9 **end**
10 **return** Segmented map;

Algorithm 1: Segmentation algorithm.

in between rooms, or rooms and corridors with different sizes. Previous work like the distance transform-based segmentation of Bormann et al. [2] used a threshold on the distance image to extract the maximum number of regions. Thus, finding a good value for the threshold depends on the size difference between the biggest and the smallest regions. Furthermore, that threshold is arbitrarily chosen by a user depending on the environment. On the other hand, our method uses the distance image to produce a new image where a pixel's value represents the size of the region it belongs to. By looking at this image, our method is independent of the shapes of regions, and only depends on the size differences *between adjacent regions*, which is a more relevant measure than the difference between the biggest and smallest regions.

In this section, we present the algorithm we use to segment maps, which is described in Algorithm 1. The key elements explained in the remainder of this section, are: A) extraction of an over-segmented map from the layout of the free space, B) removal of a particular type of region that we call ripples, C) further refining of the segmentation by merging neighbors with similar value and detecting door patterns, if any are present, D and E) removal of regions created by thick walls and contour straightening for robot maps.

A. Computation of the free space image

We first compute an image where each pixel value represents the size of the region it belongs to, referred to as the free space image (FSI). Effectively, the FSI represents the amount of free space surrounding every pixel in the image.

The algorithm to extract the FSI is shown in Algorithm 2. It starts with calculating the distance image of the map, i.e. the image where each pixel's value is the distance to its closest obstacle. For each pixel of the distance image, we create a circular mask centered on the pixel and whose radius is the value of the pixel. For every pixel in the circular mask, if the value of the equivalent pixel in the FSI is less than the circle radius, the FSI pixel's value is changed to the radius of the circle. Once all pixels have been considered, neighboring FSI pixels with the same value are grouped into regions.

Since we do not assume any particular direction for regions in the map we used a circular mask, but other shapes can be used for other application, e.g. a squared shape for environments with only orthogonal angles. An example of sketch map's FSI can be seen in Fig. 2b.
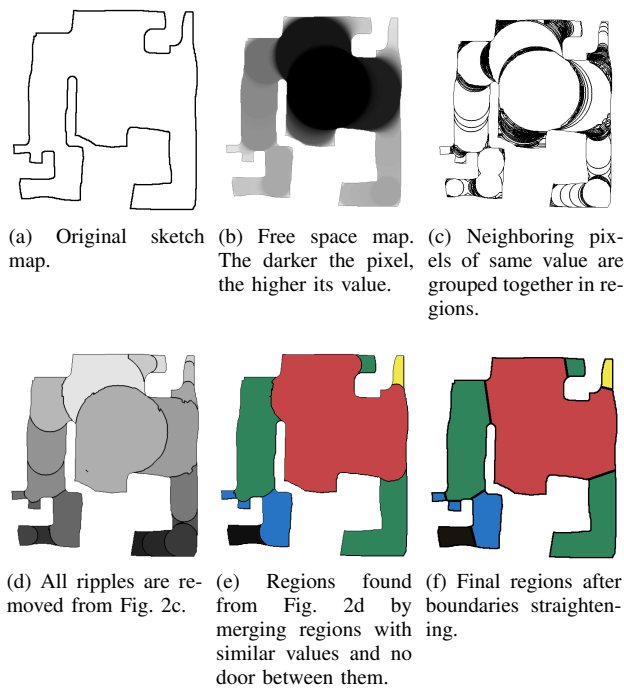
(a) Original sketch map.

(b) Free space map. The darker the pixel, the higher its value.

(c) Neighboring pixels of same value are grouped together in regions.

(d) All ripples are removed from Fig. 2c.

(e) Regions found from Fig. 2d by merging regions with similar values and no door between them.

(f) Final regions after boundaries straightening.

Fig. 2: All steps from the segmentation algorithm.

B. Merging of ripple regions

The FSI provides an over-segmented map where variations of size, however small, are represented (see Fig. 2c). Small variations in pixel value create ripples, i.e very thin regions that vary slightly in pixel values. Those are often present between bigger regions and the second step of the algorithm removes them by merging them into larger regions.

However, it is important that ripples are merged into the region they belong to. Otherwise, two ripples could be merged together, creating a non-ripple region where there shouldn't be any, over-segmenting the map. Since ripples are smaller than the region they belong to, by considering regions from the one with the highest pixel value to the lowest, the algorithm merges large ripples into large regions first, before merging the smaller ripples into the remaining regions, thus avoiding over-segmentation.

Only a simple rule is needed to detect ripples. Since ripples are elongated, around half of their contour is in contact with an adjacent region, i.e a neighbor region. If more than 40% of the contour of a region is in contact with a neighbor region, the region is merged with its neighbor. If a region is a ripple of more than one region, it is merged with the region that has the closest pixel value. Every time a ripple is merged, all neighboring regions of the ripple are checked again to see if they became ripples of the new, merged, region. A stability analysis is shown in Section IV-A confirming that using 40% leads to good segmentations of the maps.

The segmentation of a sketch map, after ripple merging, can be seen in Fig. 2d.

C. Merging of neighbor regions with similar pixel values

At this point, the map segmentation still has regions that belong to the same place, that are not ripples, but need to be

---

Data: *map image*
Result: *FSI*
1 dimage = distance image from map;
2 fsi = empty image;
3 **for** *each pixel in dimage* **do**
4   create circle with radius pixel value;
5   **for** *each pixel in circle* **do**
6   **if** *the value of the equivalent pixel in fsi $\leq$ circle radius* **then**
7   equivalent pixel in fsi = circle radius;
8   **end**
9   **end**
10 **end**
11 **return** fsi;

Algorithm 2: Free space image algorithm

merged together, e.g. corridors that slowly become smaller will be cut into multiple parts of different sizes. This is visible in Fig. 2d where the left corridor is still separated in multiple regions because its width is not constant. Thus, this step of the algorithm merges neighboring regions with similar pixel values.

We define $pV1$ and $pV2$ the pixel values of two regions, $t\_merging$ the merging threshold, and m a margin added to the merge threshold. t_mergin is a number between 0 and 1 representing the relative pixel-value difference between regions for automatic merging, with 0 being no merging possible and 1 being always merge regions. Hence, the merging threshold and margin represent how similar in value two regions need to be before being considered for merging.

We consider regions from the one that contains the most pixels to the one that contains the least, and recursively look at all neighbor regions. If the value difference $|pV1-pV2|$ is less than the highest pixel value multiplied by the threshold, i.e. Eq. (1) holds, both regions have similar pixel values and are merged.

$$|pV1 - pV2| \leq \max{(pV1, pV2)} * (t\_merging) \quad (1)$$

However, using a fixed threshold will not merge all regions that should be merged, due to the fuzzy nature of regions. Thus, for two neighboring regions, if $|pV1 - pV2|$ is more than the highest pixel value multiplied by the threshold, i.e. Eq. (1) does not hold, but it is less than the same equation with a margin m added to the threshold, i.e. Eq. (2) holds, the algorithm checks if the regions should be merged by studying their neighborhoods. It will merge the regions if and only if one of the two regions has a similar pixel value to at least one neighbor of the other region, i.e Eq. (1) holds between a region and at least one neighbor of the other region.

$$|pV1 - pV2| \leq \max{(pV1, pV2)} * (t\_merging + m) \quad (2)$$

For regions between which ripples were present, the algorithm does not consider them for merging if the ripples were created by a door. Doors can be found by looking at the minimum value of all the ripples that were present between two regions and making sure that the difference in pixel values between this minimum value and both regions was not significant, i.e. Eq. (1) holds between the minimum and both regions.
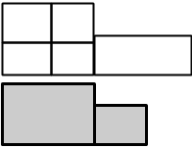
Example segmentation	Our method	Bormann's method
	$\frac{1+0.5+0+0+0}{5}$ $=\frac{1.5}{5}$ $=0.3$	$\frac{1+1+1+1+0.5}{5}$ $=\frac{4.5}{5}$ $=0.9$

Fig. 3: Difference of recall between our method and Bormann's method. The top image is the ground truth while the bottom one is the under-segmented map. With Bormann's calculation, all subregions in the biggest room increase the mean recall by being considered perfectly fitted in the biggest segmented region: four perfect recalls for one segmented region. The under-segmented room weights more on the final recall result than the corridor. With our measure, only one subregion is associated with the biggest room and all other subregions yield a score of 0, driving the recall down.

Stability analyses for $t\_merging$ and m can be found in Section IV-A. The final segmented version of Fig. 2a can be seen in Fig. 2e

D. Taking into account wall thickness

Thick walls can over-segment the map by creating small regions, e.g. where there are doors. Instead of assuming that we know the walls' thickness, the algorithm considers that every region with more than a certain percentage $d\_threshold$ of its perimeter in contact with other regions belongs to another region. Thus, those regions are fused into a neighbor region with less than $d\_threshold$ of its perimeter in contact with other regions. A stability analysis of this parameter can be found in Section IV-A.

E. Straightening the boundaries

The contact line between regions depends on the shape of the mask used in Section III-A. A circular shape works well with sketch maps since the drawing is inaccurate and boundaries between regions might not be straight. However, for metrically correct robot maps, boundaries between regions can be straightened to increase the accuracy of the segmentation. We replace each boundary between regions by the straight line between its endpoints.

While this refining step is not applied to sketch maps, for illustration purposes, a straightened segmented version of Fig. 2a can be seen in Fig. 2f.

IV. EXPERIMENTS

We compare our method (public implementation is available online[1]) with approaches presented by Bormann et al. [2] and against the DuDe method of Fermin-Leon et al. [3], using the dataset of Bormann et al. [2] and against our own dataset of sketches [10]. The dataset of Bormann et al. [2] is made of 20 maps, either without clutter or with artificially added clutter. Since our method is only adapted to environments without furniture, we used the non-cluttered images. Clutter can be removed in a preprocessing step or by extracting the walls of an environment, as done by Wulf et al. [11]; we are leaving those ideas for future work.
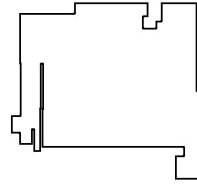

Fig. 4: Model environments of KTH SLAM dataset explored and drawn by the users to produce the sketches in the dataset.

We make available a dataset of 25 sketches. Those sketches were obtained from a virtual environment in a web browser[2] and correspond to the ground truth of the KTH dataset for SLAM[3] (Fig. 4). Each sketch map is associated with two possible ground-truth segmentations, obtained by asking two non-expert users to segment the sketches. The only instruction was to segment regions so that, if a robot or person had to visit the environment, they would have seen every part of it by visiting every region of the map.

Bormann et al. [2] introduced a quality measure based on precision and recall, where precision of a segmented region is the maximum overlapping area of the segmented region with a ground truth region divided by the area of the segmented room, while recall of a ground truth region is the maximum overlapping area of the ground truth region with a segmented room divided by the area of the ground truth region. The full segmentation's precision is the mean of all segmented regions' precision while the recall is the mean of all ground truth regions' recall. A segmentation is good if both the precision and recall are high.

While this metric evaluates how the segmentation fits the ground truth, it is biased toward giving high recall results on under-segmentation, and high precision results on over-segmentation, as illustrated for recall in Fig. 3. Indeed, the segmentation recall being the mean of the recall of all ground truth regions, one segmented region including multiple ground truth regions will have a lot of weight on the segmentation's recall. In some cases, it can virtually erase the influence of segmented regions as in Fig. 3, where the central room recalls outweigh the corridors recall. A similar case can be made for the precision.

Furthermore, since the true positive values (the pixels that fit both in the ground truth and segmented regions) are not the same for the calculation of the precision and the recall, those measures make us unable to represent a confusion matrix or calculate meaningful metrics such as the F-score, G-score or Matthews correlation coefficient.

We introduce another way to evaluate the segmentation results borrowed from clustering measures. Each segmented region is associated with the ground truth region with the biggest overlap and that is not already associated with a segmented region. The true positives (tp) are all pixels in both the segmented region and the ground truth, the false positives (fp) are the pixels in the segmented region but not in the ground truth, the false negatives (fn) are all the pixels in the ground truth region but not in the segmented, and the
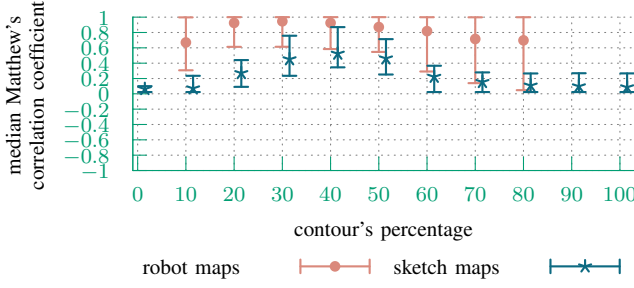
---

[1] https://github.com/MalcolmMielle/maoris

[2] http://aass.oru.se/Research/mro/smokebot/sketchmap-web/
[3] http://www.nada.kth.se/%7Ejohnf/kthdata/dataset.html

Fig. 5: Evaluation of the sensitivity of the algorithm to the parameters for fusing ripples with $t\_merging$ = 0.30 and m = 0.1. Points are the median MCC, while the whiskers represent the minimum and maximum value. When considering both map types, the best result is found when regions with 40% of their contour in contact with another region are considered ripples.
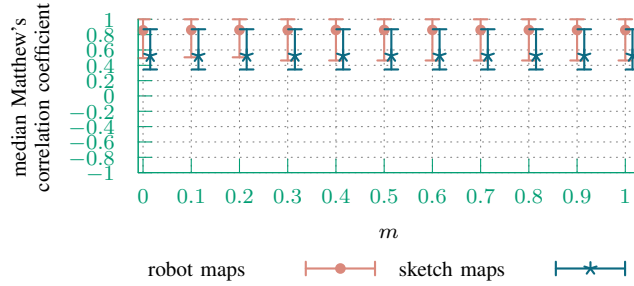

Fig. 6: Evaluation of the margin m with $t\_merging$ = 0.30. One can observe that the algorithm is largely insensitive to the value of m, but having it different than 0 results in slightly better segmentation. We obtained similar results for other values of $t\_merging$.

true negatives (tn) are all the pixels in neither the ground truth nor the segmented region. To be able to compare the different results given by each segmentation algorithm, we use the Matthews correlation coefficient (MCC):

$$MCC = \frac{tp * tn - fp * fn}{\sqrt{(tp + fp)(tp + fn)(tn + fp)(tn + fn)}} \quad (3)$$

The MCC ranges between -1 and 1, with the best predictive result being 1, 0 being no better than guessing, and -1 indicating total disagreement. One advantage of the MCC for evaluating map segmentation is that it stays balanced even when the classes are of different sizes. While there is no best way to represent a confusion matrix with a single number, the MCC is generally regarded as one of the best measures to do so [12]. Regions that were not associated with a corresponding region have a MCC of 0. The final MCC of a segmented map is the mean of all regions' MCC. A public implementation of the evaluation program can be found online with the MAORIS package.

A. Evaluation of the parameters' stabilities

We did a stability analysis of every parameter of our method using median MCC as a measure of segmentation goodness. We ran the analyses over the 16 smallest uncluttered maps of Bormann's dataset and all maps in our sketch dataset against one of the user's ground truth segmentation.
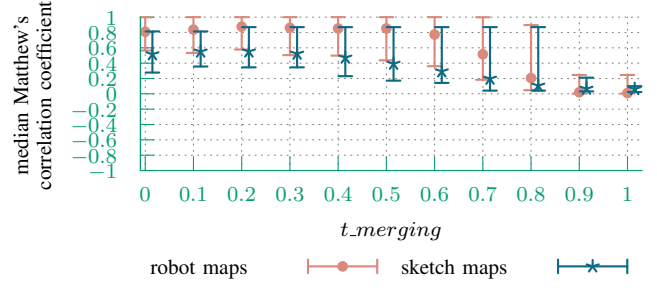
Fig. 7: Evaluation of $t\_merging$ with $m = 0.1$. The best results are found between 0 and 0.3 for sketch maps and between 0.1 and 0.5 for robot maps. Thus a good value of $t\_merging$ for both would be 0.3.
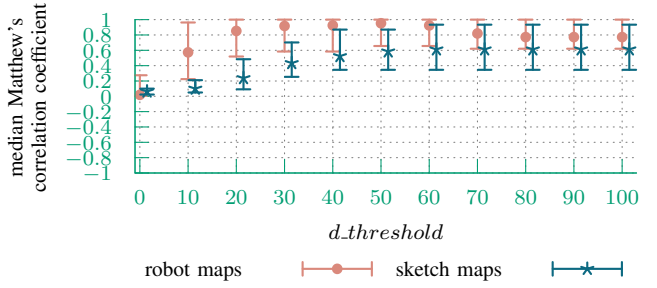

Fig. 8: Evaluation of the sensitivity of the algorithm to the parameters for fusing regions created by thick walls with $t\_merging$ = 0.30 and m = 0.1. Sketch maps' segmentations' results do not depend much on this parameter: they are good above 40%, with the highest score at 60%, and a very small decrease until 100% where $d\_threshold$ is not used. However, it is an important factor for robot maps: the best results are found between 15 and 60% with a sharp decrease of the median Matthews correlation coefficient after.

We first confirmed that using 40%, as the threshold value above which a region is determined to be a ripple, was a valid assumption. As can be seen in Fig. 5, the MCC is best between 30% and 45%, with the highest value at 40%, confirming the validity of the threshold's value.

We then evaluated the segmentation results' stability to $t\_merging$, the threshold for fusing adjacent regions with similar pixel values, and the margin m. As one can see in Fig. 6, the algorithm is not sensitive to the value of the margin m for it has no influence on the sketch maps segmentation and only slightly increases the robot maps' segmentation's MCC when different than 0. In our work, we chose to have $m = 0.1$. The analysis for the threshold $t\_merging$ can be seen in Fig. 7. One can see that the segmentation goodness increases with $t\_merging$ but decreases passing a certain point: around $t\_merging$ = 0.5 for the robot maps and $t\_merging$ = 0.3 for the sketch maps. Also, one should note that the MCCs over the robot maps is constant between $t\_merging$ = 0.2 and 0.4. Those different values between sketch and robot maps come from the presence of doors in between regions in robot maps. Doors create a separation between regions, even when they are of similar size, allowing the merging threshold to be higher without merging regions that shouldn't be merged. However, sketch
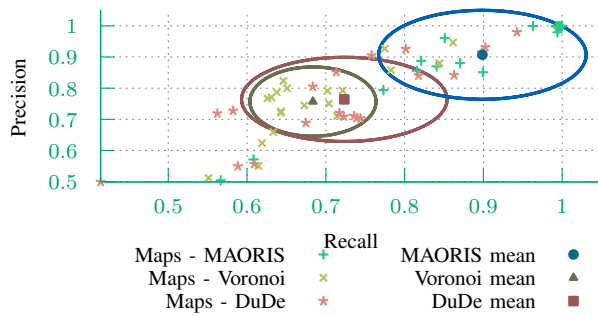
Fig. 9: Precision and recall for every unfurnished map in Bormann et al. [2] dataset. One can see that MAORIS outperforms the Voronoi segmentation and DuDe both in precision and recall. MAORIS has a median MCC of 0.98, while Voronoi has 0.65 and DuDe has 0.70.

Dataset	MAORIS	Voronoi	DuDe
Bormann's	0.98	0.65	0.70
Sketch	0.56	0.28	0.30

TABLE I: Median Matthews correlation coefficient for each method on all maps of both datasets. The closer this coefficient is to 1 the better the segmentation is.

maps in our dataset have no doors, leading to lower median MCC values above $t\_merging = 0.3$. To get good results for both sketch and robot maps, we chose $t\_merging = 0.3$. Hence, regions between which the difference of size is more than 1/3 of the size of the biggest region are not merged together.

Finally, we evaluated the influence of $d\_threshold$ on the segmentation results. One can see in Fig. 8 that sketch map segmentations are not sensible to this parameter: the median MCC of the sketch maps segmentations is good above 40% and its value increases until 60% after which it only slightly decreases until 100%. We hypothesize that users took into account the size of the pen while drawing, and thus, there are no regions created by the wall thickness. On the other hand, $d\_threshold$ is an important parameter for robot maps: the best results are obtained between 30% and 60%, and stays correct until 100%. For applications where one doesn't know if the wall thickness needs to be taken into account or not, we recommend 40% as the best guess. In our work, we chose $d\_threshold$ to be 40% for robot maps and 60% for sketch maps.

B. Segmentation results on Bormann's dataset

We ran our algorihtm (MAORIS) on the 20 uncluttered maps proposed by Bormann et al. [2]. The result in precision and recall for each map can be seen in Fig. 9 and MCCs are shown in Table I. The Voronoi based segmentation of Bormann et al. [2] obtained an MCC of 0.65, and the DuDe method of Fermin-Leon et al. [3] obtained an MCC of 0.70. In comparison, our method got a median MCC of 0.98, thus performing better than the two other methods.

Examples of segmentations given by all three algorithms on maps from Bormann's dataset can be seen in Fig. 11.
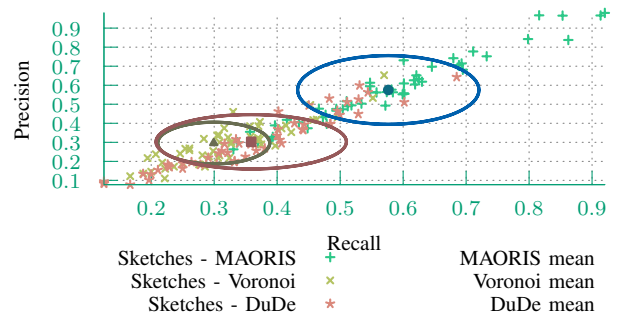

Fig. 10: Precision and recall for each sketch in the dataset using batch DuDe, Voronoi segmentation, and our method for both evaluated using two ground truths. Ellipses represent the standard deviation on the recall and precision for each distribution. Our method outperforms the other methods with a median MCC of 0.56, while Voronoi has 0.28 and DuDe has 0.30.

C. Segmentation results on our dataset of sketch maps

We ran the three algorithms on 25 sketch maps and two user-provided ground truths. MAORIS performs better at segmenting sketch maps, with an MCC of 0.56, compared to the Voronoi segmentation with an MCC of 0.28 and the DuDe method with an MCC of 0.30. When looking at Fig. 10, one can see that both the Voronoi segmentation and DuDe have high precision and low recall. On the other hand, MAORIS has a better balance of precision and recall.

Taking into account that sketch maps in our dataset do not have doors, it should also be noted that, with an MCC of 0.56, MAORIS does not depend on doors between regions for segmentation, as opposed to previous work [5].

Examples of segmentations given by all three algorithms on maps from our dataset of sketches can be seen in Fig. 11.

V. LIMITATIONS AND FUTURE WORK

MAORIS' speed depends on the image size. Since the method is pixel based, the more pixels in the image, the slower the method. This drawback can be easily overcome by reducing the resolution of large images. Over the datasets of Bormann et al. [2], MAORIS has a mean processing time of 43s when run on an Intel i7-4712HQ at 2.30GHz with 16GB of RAM. However, most of the processing time comes from 2 large maps. Without those 2 maps, the average processing time is 8.8s. Over the Bormann's dataset, with an Intel Core i7 CPU at 2.70GHz, the DuDe method processed 95% of the maps in under 1s [3] and the Voronoi segmentation took about 13s [2]. Over the sketch dataset MAORIS has a mean processing time of 6s. Developing an incremental method of segmentation for MAORIS would increase the processing speed. We leave this for future work.

Another way MAORIS could be improved is by being able to handle clutter in maps. In the future, we plan to work on ways to automatically remove clutter from maps before segmentation, allowing us to segment cluttered maps. We also plan to use regions given by the MAORIS segmentation method to perform matching between maps from different modalities.

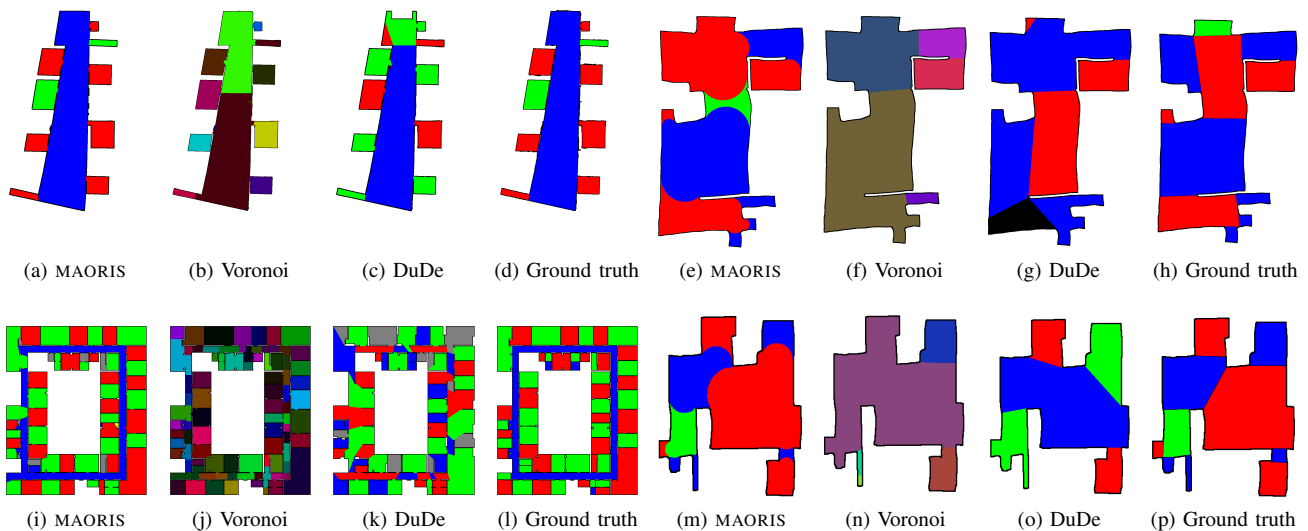|							
(a) MAORIS	(b) Voronoi	(c) DuDe	(d) Ground truth	(e) MAORIS	(f) Voronoi	(g) DuDe	(h) Ground truth
(i) MAORIS	(j) Voronoi	(k) DuDe	(l) Ground truth	(m) MAORIS	(n) Voronoi	(o) DuDe	(p) Ground truth

Fig. 11: Examples of segmentation results for all algorithms and associated ground truths. Left: maps from Bormann's dataset. Right: maps from our dataset of sketches. One can see that MAORIS performed better than the other methods. For example the corridor in Fig. 11i is almost entirely extracted as one region while the two other methods over segment it.

VI. SUMMARY

We developed a strategy to segment maps from different modalities, and evaluated it on robot built and sketch maps. While robot maps are metrically accurate representations of an environment they suffer from noise in the measurements. On the other hand, sketch maps represent the topology of the environment but are not metrically accurate and contain inaccuracies, sometimes even done purposely by the user.

We also identified a flaw in the way segmentation have been evaluated in recent works and proposed a more consistent metric based on Matthew's correlation coefficient.

We evaluate our algorithm against state of the art algorithms for map segmentation, on two datasets: one standard benchmark dataset of robot maps, and one dataset of hand-drawn sketch maps that we make publicly available. Our method outperforms the state of art algorithms and obtains better Matthew's correlation coefficients for both.

REFERENCES

[1] M. Skubic, D. Anderson, S. Blisard, D. Perzanowski, and A. Schultz, "Using a hand-drawn sketch to control a team of robots," *Autonomous Robots*, vol. 22, no. 4, pp. 399–410, Mar. 20, 2007, ISSN: 0929-5593, 1573-7527. DOI: 10.1007/s10514-007-9023-1.

[2] R. Bormann, F. Jordan, W. Li, J. Hampp, and M. Hägele, "Room segmentation: Survey, implementation, and analysis," in *IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2016, pp. 1019–1026.

[3] L. Fermin-Leon, J. Neira, and J. A. Castellanos, "Incremental contour-based topological segmentation for robot exploration," in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2017, pp. 2554–2561. DOI: 10.1109/ICRA.2017.7989297.

[4] G. Liu, Z. Xi, and J.-M. Lien, "Dual-space decomposition of 2d complex shapes," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 4154–4161.

[5] E. Fabrizi and A. Saffiotti, "Extracting topology-based maps from gridmaps," in *IEEE International Conference on Robotics and Automation (ICRA)*, vol. 3, IEEE, 2000, pp. 2972–2978.

[6] J. Park, A. J. Sinclair, R. E. Sherrill, E. A. Doucette, and J. W. Curtis, "Map merging of rotated, corrupted, and different scale maps using rectangular features," in *2016 IEEE/ION Position, Location and Navigation Symposium (PLANS)*, Apr. 2016, pp. 535–543. DOI: 10.1109/PLANS.2016.7479743.

[7] S. Ahmed, M. Liwicki, M. Weber, and A. Dengel, "Automatic room detection and room labeling from architectural floor plans," in *Document Analysis Systems (DAS), 2012 10th IAPR International Workshop on*, IEEE, 2012, pp. 339–343.

[8] L.-P. de las Heras, S. Ahmed, M. Liwicki, E. Valveny, and G. Sánchez, "Statistical segmentation and structural recognition for floor plan interpretation: Notation invariant structural element recognition," *International Journal on Document Analysis and Recognition (IJDAR)*, vol. 17, no. 3, pp. 221–237, Sep. 2014, ISSN: 1433-2833, 1433-2825. DOI: 10.1007/s10032-013-0215-2.

[9] A. Diosi, G. Taylor, and L. Kleeman, "Interactive SLAM using laser and advanced sonar," in *IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2005, pp. 1103–1108.

[10] M. Mielle, M. Magnusson, and A. J. Lilienthal, *Sketch maps dataset*, Sep. 2017. DOI: 10.5281/zenodo.892062.

[11] O. Wulf, K. O. Arras, H. I. Christensen, and B. Wagner, "2d mapping of cluttered indoor environments by means of 3d perception," in *IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, vol. 4, 2004, pp. 4204–4209.

[12] D. M. Powers, "Evaluation: From precision, recall and f-measure to ROC, informedness, markedness and correlation," Tech. Rep., 2011.