Meta-rooms: Building and Maintaining Long Term Spatial Models in a Dynamic World

Rareș Ambruș[1], Nils Bore[1], John Folkesson[1] and Patric Jensfelt[1]

***Abstract*— We present a novel method for re-creating the static structure of cluttered office environments - which we define as the "meta-room" - from multiple observations collected by an autonomous robot equipped with an RGB-D depth camera over extended periods of time. Our method works directly with point clusters by identifying what has changed from one observation to the next, removing the dynamic elements and at the same time adding previously occluded objects to reconstruct the underlying static structure as accurately as possible. The process of constructing the meta-rooms is iterative and it is designed to incorporate new data as it becomes available, as well as to be robust to environment changes. The latest estimate of the meta-room is used to differentiate and extract clusters of dynamic objects from observations. In addition, we present a method for re-identifying the extracted dynamic objects across observations thus mapping their spatial behaviour over extended periods of time.**

I. Introduction

Robots are operating for longer times and collecting much more data than just a few years ago. We are currently studying long term unsupervised learning by robots operating for months at a stretch, which results in many challenges that need to be addressed. For instance, the handling of the huge amounts of data and extracting the interesting and relevant parts of that data is critical. In this paper we develop a framework for doing just that by separating the static parts of a room scene from the dynamic ones and improving this separation as the robot returns to the room many times.

By being able to identify these static parts the robot can better localize itself and navigate the environment. It can also detect when a change in the environment is 'normal', that is, only dynamic objects have moved or when it is anomalous, that is, something that was believed to be static has now moved. This is important in, for example, a watchman robot scenario.

The data from dynamic parts of our environment properly registered and segmented will be valuable as input to object classifiers to find out what they might be or to initialize new object classes. The motions of these objects can then be recorded and analysed for patterns leading to more complete models of them. Unusual motion can then also be detected as an another kind of anomaly. All of this is outside the scope of this paper but relies on a good way to recursively estimate the dynamic/static separation in our 3D sensor data.

In this paper we present a method that allows an autonomous robot to iteratively learn to distinguish between static and dynamic parts in the environment. In particular, we focus on the problem of reconstructing the static structure of the environment and we analyse the stability and convergence of our approach to iteratively rebuilding the static structure from partial observations. Finally, we show that using our method we can successfully extract dynamic objects which can further be matched across observations.

[1] The authors are all with the Centre for Autonomous Systems at KTH Royal Institute of Technology, Stockholm, SE-100 44, Sweden. {raambrus, nbore, johnf, patric}@kth.se

II. Related work

The problem of long term operation in dynamic environments has received a fair amount of attention in the robotics community. In the context of SLAM on 2D grid maps, the study of dynamics has lead to complex formulations of the traditional, static grid cell occupancy that take into account, in addition to occupancy, the dynamic element. For example, Saarinen et al use an independent Markov chain to model dynamics within grid cells [1]; Kuchner et al extend the traditional grid map approach with a Conditional Transition Map which models, for each grid cell, the transition probability of a dynamic object moving to one of its neighbouring cells, using a roundabout as a test scenario [2]. Walcott-Bryant et al. address the problem of changing environments in a Pose-Graph SLAM system by augmenting a node in the graph with information concerning its dynamic state, and they maintain two separate maps - active and dynamic - to better represent the environment [3]. Bieber and Duckett maintain the traditional occupancy grid cell definition but they employ a number of maps, called sample maps, which adapt to changes at different rates and time scales [4]. The Normal Distribution Transform (NDT) approach to grid map representation has also been applied successfully for the detection of dynamic cells in 3D, by Andreasson et al. in the context of an autonomous security patrol robot [5] and, more recently, by Saarinen et al. in complex and highly dynamic environments [6]. Our approach to dealing with dynamic environments over large periods of time takes a significant paradigm shift from traditional grid based methods of representing occupancy and dynamics. Instead, we focus on well-localized clusters of points, and we reconstruct the environment based on their motion (or the absence of it).

At the semantic map level, Gunther et al. build large scale 3D maps from RGB-D data which they further augment with recognized objects based on clustered planar regions; finally they refine the map by replacing the objects with their corresponding CAD models [7]. Mason et al. describe experiments over extended periods of time in which they build semantic maps for object query as well as change

detection, and they focus on objects lying on top of tables and other planar surfaces; the matching of these objects is done on the basis of the overlap between their convex hulls in 2D [8]. In contrast, our approach does not actively look for objects, but instead we focus on reconstructing the static structure of the environment, on the basis of which the dynamic objects can be easily obtained.

Another way of looking at the problem of dynamics in the environment is by actively searching the scene for objects which are known to be dynamic, such as chairs, tables, cups, etc. The problem of cluster segmentation and object recognition and/or reconstruction from RGB-D data has become a very active topic, especially with the advent and wide availability of low cost sensors capable of recording such data with high accuracy. There is a wide body of work that deals with recognition in a single view / single scene, however, as this is not the primary aim of our work we will focus on some of the more relevant methods which deal with combined data from multiple scenes. Koppula et al. fuse individual RGB-D frames into larger point clouds from which they extract and label objects via a graphical model based on individual object features as well as inter-object relationships in the form of proximity, visual similarity, etc; they demonstrate their system in an online experiment where the robot is able to actively search and re-identify these objects in cluttered scenes [9]. Ruhnke et al. take a different approach to object reconstruction: the input to their method is a set of of 3D laser range scans on which they perform plane segmentation to remove the floor and walls and subsequently form clusters from the remaining points; further they compare clusters across different observations and use range images to obtain likely matches, after which they proceed to merging the clusters into objects [10]. We take a similar approach, however, we deal with much larger spaces and we use partial clusters for the reconstruction of complete rooms as opposed to focusing on individual objects. Within the sphere of object recognition, the works closest to our method are the ones that rely on change detection across observations. The method proposed by Herbst et al. in [11] and further refined in [12] uses surface patches to describe a scene (usually of a table top with various objects), and they develop a sophisticated method of tracking and re-identifying the surface patches across observations. Along similar lines, Alimi et al. use scene differencing for both object extraction as well as for modelling the static scene background [13]. Finman et al. also use change detection for object extraction [14], however their method is geared towards dealing with large RGB-D maps, and they use differencing to filter out the static structure and extract point clusters which are later matched in new observations of the map. Beuter et al. use scene differencing to extract static, moving and movable entities [15] in the vista space (short observations of some part of the world from the same view-point).

Our method relies on scene differencing for static structure reconstruction, however, we differ from the approaches in [11], [12], [13], [14], [15] in a few key points: our work environment is much larger, we deal with whole rooms as opposed to table tops, and we perform our experiments over longer periods of time; in addition to incorporating occluded clusters (which we determine via a more capable method than that described in the papers cited) into the static structure, our method has the capability of adapting to other objects that have been introduced in the environment, such as additional pieces of furniture; and finally, we perform an analysis on the convergence and stability of the static structure created.

III. System Overview

Our system consists of the robot collecting a sequence of RGB-D frames from a room and registering them in a common reference frame. This dense point cloud over the contents of the room is down sampled and filtered to remove outliers. The data is compared to our current model of the static room which we call a meta-room. The differences between them are analysed to see what should be added or removed from the meta room and what parts of the current data are dynamic. In doing so care must be taken to consider the effect of occlusions and of the noise inherent in the sensor. It is important that parts of the meta-room that appear dynamic due to noise are not removed. At the same time, parts that were previously occluded but now can be seen need to be added. The meta-room should be able to change if there is repeated evidence of a new static part in the room.

After we have a good stable meta-room it is possible to analyse the dynamic parts of each point cloud to find dynamic clusters of points. By registering these clusters from different visits to the room we can build up meta-object models of, for example, chairs that are always present but moving between visits.

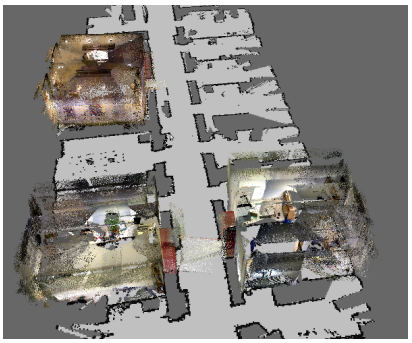


Fig. 1: Registered scans of different office rooms.

Fig. 1 shows the resulting registered point clouds of three different rooms on the 2D map, each point cloud consisting of about 2 million points. In addition to the method described in the next sections, we contribute the collected dataset.

IV. Meta-rooms

As said, a meta-room consists of those parts of a room which have been observed as static over time, and we construct one meta-room for each different room. Fig 2 shows different observations of the same room, and it also serves the purpose of exemplifying some of the issues that need to be addressed in order to reconstruct the static structure of a

(a) (b) (c) (d)

Fig. 2: Four observations of the same office room at different times. The red ellipses in b) and c) indicate missing data due to sensor error and due to occlusions.

room accurately (the ceiling points have been removed for better visibility). The red ellipse in fig 2 b) indicates an area where data is missing due to sensor error - the cupboard on the right. Second, the issue of occlusions: as objects move around, whatever part of the environment is behind them is occluded to the depth camera; this can be seed in Fig 2 c) where the red ellipse shows part of the couch that is missing as it is occluded by the chair in front of it.

A. Initialization

A meta-room is constructed through an iterative process, taking into account new room observations when they are available. In the beginning the meta-room is empty and it is initialized with the first room observation. The second step performed in the initialization phase is to detect bounding primitives, using a RANSAC based estimator; this is primarily used to filter out points that lie outside the room (i.e. points observed through windows or doors).

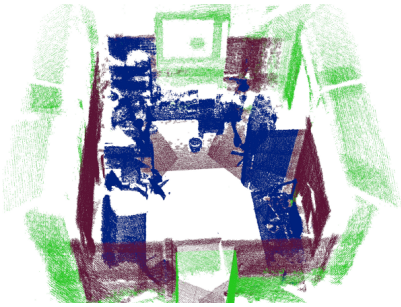


Fig. 3: Room observation segmented into bounding primitives (red), interior points (blue) and points outside the room (green). The ceiling points have been removed for better visibility. Best viewed in color.

The algorithm we have developed first extracts the floor and ceiling primitives, the heights of which we determine through a histogram analysis of the heights of all the points in the point cloud and assuming that the vertical direction is known. For detecting the bounding primitives we first compute all the primitives perpendicular to the floor primitive, and subsequently through simple geometric operations we find the ones on the "outside" (i.e. the ones for which all other primitives lie only on one side, in 3D space) - as shown in Fig 3. The rooms involved in our experiments are square and have planar surfaces as boundaries, but the method can deal with different shaped rooms which don't necessarily have planar walls, the limitation currently being that we can only handle convex room shapes.

B. Update

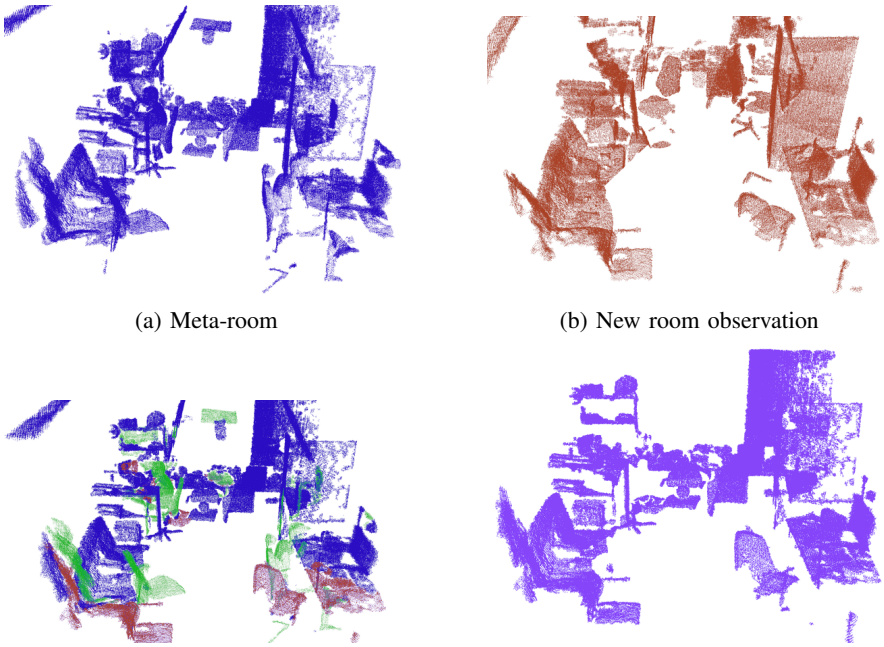
After initialization, the meta-room contains both static and dynamic elements. The update step is twofold: the dynamic elements will be removed while the space occluded by them will have to be added, as it could be a previously obscured part of the static structure. Whenever a new room observation is available, it is first registered with the meta-room (i.e. transformed into the meta-room frame of reference); although both the meta-room and the new room observation are acquired by a robot which is already localized on the 2D map, this registration step is required because the underlying localization module assumes a static environment and suffers from small errors which would propagate through the rest of the pipeline. For the registration of complete room point clouds we use the Normal Distribution Transform (NDT) algorithm [16] which in our experiments outperforms the Iterative Closest Point (ICP) algorithm [17]. ICP, however, performs better when aligning individual point clouds (described in more detail in section V).

Once the point clouds have been aligned we update the meta-room by looking at the differences between the two point clouds. We define the difference operator $\ominus$ such that $S = P \ominus Q$, where P and Q are point clouds, results in the point cloud S given by:

$$S = \{p \mid p \in P \ \wedge \ \forall q \in Q, \ ||p,q|| > d\} \qquad (1)$$

where d is a threshold, which in our experiments is set to 1 cm. Thus the set S contains all the points of the first point cloud that don't have a nearest neighbour within the specified threshold in the second point cloud. Note that in our context this operation can only be performed on aligned point clouds.

1) Handling occlusions: The point cloud representing the difference between the meta-room and the new room observation contains all the points which are present in the meta-room and are not present in the new room observation.



(a) Meta-room

(b) New room observation

(c) Meta-room (blue), clusters to be removed (green) clusters to be added (red)

(d) Updated meta-room

Fig. 4: Meta-room update process. Best viewed in color.

However, it cannot immediately be concluded that all these points should be removed from the meta-room, as some of them might be part of the static structure and are simply occluded by other points in the new room observation. In fact, every dynamic part in the meta-room results in an occluded part in the room and vice-versa, and this needs to be taken into account during the meta-room update process.

Fig 4 shows an example of a meta-room (subfigure a) and a new room observation (subfigure b). Clearly some things are dynamic and should be removed from the meta-room, and at the same time, parts of the static structure present in the new room observation, like the desk, should be added. Before dealing with occlusions we first cluster the differences using a connected components analysis of the point cloud, where all the points belonging to the same component are within a certain distance of each other. This allows us to reason about objects, or at least parts of objects, and also helps to remove some points corresponding to noise.

Algorithm 1 outlines the method we employ to update the meta-room. We first take the point cloud difference (as defined in equation 1) both ways (i.e. from the room to the meta-room and from the meta-room to the room), we cluster both difference point clouds and we subsequently check both sets of clusters for occlusions (with C_1 being the set of clusters from the meta-room to room difference point cloud and C_2 being the set of clusters from the room to meta-room difference point cloud):

- If a cluster from C_1 is occluded by another cluster from C_2, it should not be removed from the meta-room, as it could be part of the static structure.
- If a cluster from C_2 is occluded by another cluster from C_1, it should be added to the meta-room, as it could be part of the static structure.

Algorithm 1 Check occlusions (mr - metaroom, r - room)

```
1:  procedure CHECKOCCLUSIONS(mr, r)
2:      S1 ← mr ⊖ r
3:      S2 ← r ⊖ mr
4:      C1 ← connectedComponents(S1)
5:      C2 ← connectedComponents(S2)
6:      toRemove ← ∅
7:      toAdd ← ∅
8:      for c1 ∈ C1 do
9:          for c2 ∈ C2 do
10:             if ¬ occluded(c1,c2) then toRemove += c1
11:             end if
12:             if occluded(c2,c1) then toAdd += c2
13:             end if
14:         end for
15:     end for
16:     mr += toAdd
17:     mr −= toRemove
18: end procedure
```

To check whether one cluster occludes another cluster (operation performed on lines 10 and 12 of algorithm 1, we project both clusters on a sphere located in the center of the room (i.e. the position of the depth camera when the data was recorded - explained in more depth in section V).

Figure 4 c) shows the clusters to be removed from the meta-room in green and the clusters to be added in red, while figure 4 d) shows the updated meta-room after this iteration.

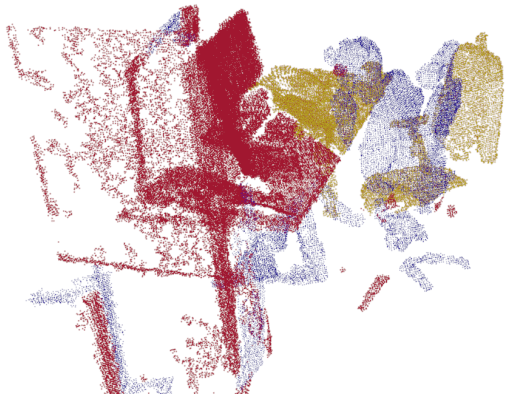


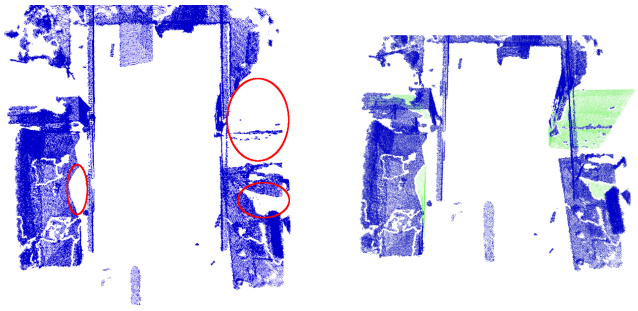
Fig. 5: Meta-room points (red), points to be removed (blue), points to be added from a new room observation (yellow). Best viewed in color.

Figure 5 shows a part of the meta-room point cloud during the update process: the points in blue are meta-room points classified as dynamic and are to be removed, while the points in yellow belong to the new room observation and are considered to be occluded and are to be added. However, the points in yellow also contain the chair, which is in fact dynamic, but it appears as (partially) occluded by the cluster of points previously present in the meta-room (the person on the chair). This is a direct result of the way we reason about occlusions, however in the long run, as the chair is indeed dynamic it moves around and it will be eliminated altogether.

C. Dealing with room changes

Another situation that needs to be handled involves changes in the environment which do not fall in the category of dynamic objects (e.g. the addition of a new cupboard in a room, or changing the position of a desk), and the meta-room needs to be able to reflect these changes accordingly. In order to account for these changes, we have developed a statistical component that identifies, for a given meta-room, static clusters of points that have been left out. To achieve this we employ a sliding window over past room observations used to update the meta-room, we take the point cloud difference between the room observations and the meta-room as defined in equation 1 and we look for similar clusters which are present in most of the differences. The reasoning is that if a cluster is not in the meta-room but it is present in all the room differences considered, and in the same position spatially, it is a static cluster and it should be added to the meta-room. To find these "similar" clusters, we use a kd-tree on the centroids of the clusters to find the ones that have very close neighbours across the room observations (the threshold in our experiments is 5 cm, thus accounting for small alignment errors), and we use the Visual Feature Histogram (VFH) [18] on the matched clusters to make sure they belong to the same class.

Future experiments will be performed over much longer periods of time (up to six months), and we will modify our sliding windows algorithm to take into account the temporal distribution of the observations, i.e. we will differentiate between observations which are minutes and observations which are days apart.



(a) (b)

Fig. 6: Meta-room vs updated meta-room with sliding window clusters. Best viewed in color.

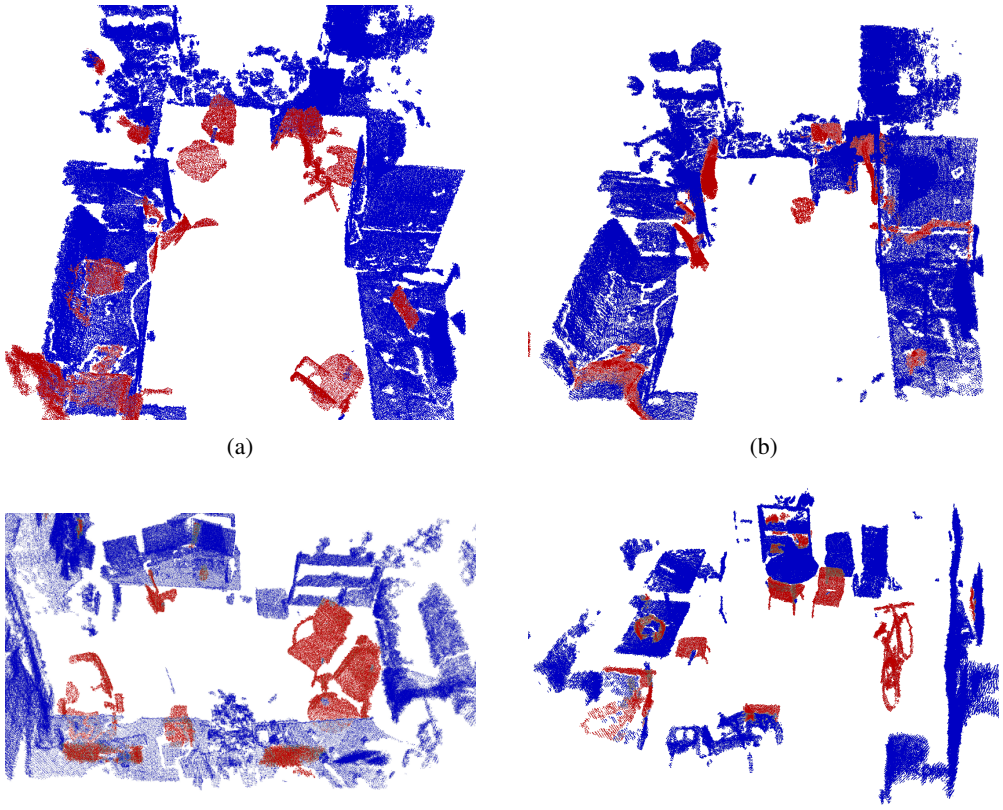
This method also helps correct some inconsistencies in the meta-room, as for example the absence of a cluster which has been eliminated (or hasn't been added) due to an alignment error when updating the meta-room with a new observation. Fig.6 shows an instance where the sliding window algorithm has identified static clusters that should be a part of the meta-room (the cupboard, part of the couch and part of the desk, shown in green in fig. 6 b).

V. Experimental Setup

Fig. 7: Scitos G5 Robot

Fig. 8: 2D map with waypoints

The data used for the experiments presented in this paper is collected by an autonomous Scitos G5 robot conducting patrol runs in an office environment, over the course of one week. A 2D map of the office is built in advance using Gmapping [19], and the robot navigates through a set of predefined waypoints while using a probabilistic particle filter based method (AMCL [20]) for localization on the 2D map. The robot (displayed in fig. 7) is equipped with an Asus



(a) (b) (c) (d)

Fig. 9: Meta-rooms with dynamic clusters. Best viewed in color.

Xtion depth camera mounted on top of a pan tilt unit at a height of 1.60 meters above the ground. The navigation waypoints have been chosen in the center of three different office rooms (as shown in fig. 8), and the robot performs an autonomous patrol run at a certain time every day and collects data inside the rooms using the depth camera.

After reaching a waypoint, the robot stops moving while the pan-tilt unit executes a sweep, pausing for one second at a number of positions while the sensor records approximately 15 frames (for filtering); for our experiments we used increments of 60 degrees horizontally and 30 degrees vertically, resulting in a total of 28 different positions of the pan-tilt which cover the room entirely, except for parts of the floor and the ceiling which are directly above and below the robot. The data acquired at each individual position is first filtered (and downsampled) through a voxel grid with a leaf size of 1 cm to remove some of the sensor noise, and then registered together using the ICP algorithm. The robot position as given by the AMCL localization combined with the pose of the pan-tilt unit gives a good initial estimate for the pose of each individual scan, and the ICP refinement removes almost all the alignment errors. Further, to eliminate outliers and spurious points in the resulting registered point cloud we run a statistical outlier removal procedure based on the mean and standard deviation of neighbouring points [21].

VI. Results

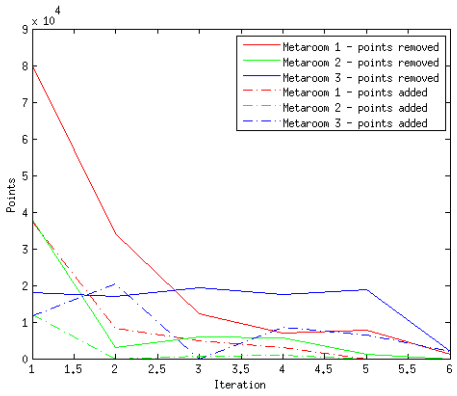


Fig. 10: Analysis of meta-room convergence with respect to points removed / added over all iterations

Fig. 10 shows the progression of all the meta-rooms over the update iterations; after a while the meta-rooms converge and the number of points added or removed approaches zero.

Once the meta-room has converged to a stable state, it can be used effectively to extract dynamic objects, by subtracting it from new observations. Fig. 9 displays a few examples of the meta-rooms and the dynamic objects detected.

TABLE I: Percentage difference in meta-rooms created from random sequences of observations of the same room. The point cloud difference is obtained using a distance threshold of 1 cm. The average meta-room difference is 2.2 % of its point clouds size.

	Seq 1	Seq 2	Seq 3	Seq 4	Seq 5	Seq 6	Avg
Seq 1 (%)	0	0.5	0.5	1.0	0.5	3.8	1.2
Seq 2 (%)	0.7	0	0.7	1.1	1.1	5.6	1.9
Seq 3 (%)	0	4.1	0	3.7	0	2.9	2.1
Seq 4 (%)	2.8	2.9	2.9	0	2.5	6	3.4
Seq 5 (%)	0	3.5	0	3.5	0	2.6	1.9
Seq 6 (%)	1.2	6.1	1.3	4.8	0.7	0	2.8

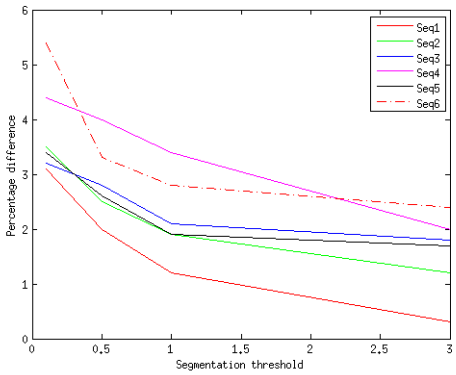


Fig. 11: Analysis of meta-room difference with various difference thresholds (in cm).

(a) Chair (b) Pillow (c) Person on chair

(d) Laptop screen (e) Chair (f) Lamp

(g) Backpack (h) Bicycle (i) Chair

Fig. 12: Examples of dynamic clusters

To check the consistency of the meta-room creation method, we have run an offline test in which we process observations of a room in random order. We generated six such random sequences, processed the observations and created six different meta-rooms which we then compared against each other; the comparison between two meta-rooms is based on the difference between their point clouds, and we quantify the error as the ratio between the size of the difference (in terms of number of points) and the size of the meta-room, with 0% being identical and 100% being completely different. The results are show in table I, with the average difference between meta-rooms being 2.2%. This difference is primarily due to sensor noise, especially for objects close to the wall the furthest away from the sensor, where the accuracy of the depth camera goes down and second due to the misalignment of the observations during the NDT registration phase. Fig. 11 shows the trend in meta-room difference when varying the difference threshold, between 0.1 cm and 3 cm. As expected, when increasing the threshold the average difference percentage goes down, which further reinforces our conclusion that the minute differences between the computed meta-rooms are mostly due to small misplacements in the point cloud due to sensor noise.

Fig. 12 shows that the current method can be applied to successfully extract and cluster various dynamic objects. Currently, we are able to detect objects that include chairs of various sizes and shapes, backpacks, monitors, jackets, lamps and up to objects as small as laptop screens. We plan to extend the method to be able to distinguish smaller objects such as cups, particularly on table tops, however, currently we are limited in this respect by the down-sampling and noise filtering techniques we employ.

Fig. 13: Sets of clusters matched across observations (dark green, light green, yellow and red) rendered alongside the meta-room (blue). Best viewed in color.

Matching objects across observations is inherently a hard

problem due to the fact that the objects are rarely seen from the same angle and that usually they are only partially seen. We have implemented a simple matching algorithm that takes into account a few features, such as space proximity (we have taken a radius of one meter around a given cluster), similar point clouds size and a few other characteristics such as height from the ground and cluster width; we also use VFH for further disambiguation. Once we have matched clusters we are able to map their position in time, across observations. Fig. 13 shows an example of matched clusters, with each set of matched clusters rendered in a different color.

VII. Conclusion and future work

In this paper we presented a method that allows an autonomous robot to distinguish between static and dynamic objects as it learns about its environment over extended periods of time and without any human supervision or intervention. In particular, we have made no prior assumption about the environment and we have focused on a purely point cloud based approach to detecting and separating dynamic elements from static ones. In addition, we show that it is possible to reconstruct the static structure of the environment from multiple observations where parts of the environment might be occluded or missing, and that our method can cope with changes in the environment such as the addition of furniture. We further show that, once we have reconstructed the static structure of the environment, we can use it to detect clusters of dynamic objects in new observations, and that these clusters can be re-identified across observations, thus mapping out their spatial distribution in time. In addition to the method presented, we will make the collected dataset consisting of 21 registered room observations (588 individual RGB-D scans) publicly available.

The method described presents us with a few avenues for further research: first, we will modify the underlying localization module, using the created meta-room as a feedback system to indicate which parts of the environment are static and thus good candidates for localization features. Second, we will focus on improving the cluster matching algorithm across observations, and we will try to reconstruct the objects from partial data. Our next goal is the creation of meta-objects, which would result from the matching of these partial clusters as they are observed across observations. And finally, we will use the system to estimate object likelihoods both spatially and temporally, and use this to detect anomalies in the environment (e.g. fig. 9 d, bicycle in the kitchen should clearly be flagged as an anomaly).

VIII. Acknowledgements

The work presented in this papers has been funded by the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement No 600623 (”STRANDS”), the Swedish Foundation for Strategic Research (SSF) through its Centre for Autonomous Systems and the Swedish Research Council (VR) under grant C0475401.

References

[1] J. Saarinen, H. Andreasson, and A. J. Lilienthal, “Independent markov chain occupancy grid maps for representation of dynamic environment,” in *IROS, 2012 IEEE/RSJ International Conference on*. IEEE, 2012, pp. 3489–3495.

[2] T. Kucner, J. Saarinen, M. Magnusson, and A. J. Lilienthal, “Conditional transition maps: Learning motion patterns in dynamic environments,” in *IROS, 2013 IEEE/RSJ International Conference on*. IEEE, 2013, pp. 1196–1201.

[3] A. Walcott-Bryant, M. Kaess, H. Johannsson, and J. J. Leonard, “Dynamic pose graph slam: Long-term mapping in low dynamic environments,” in *IROS, 2012 IEEE/RSJ International Conference on*. IEEE, 2012, pp. 1871–1878.

[4] P. Biber and T. Duckett, “Experimental analysis of sample-based maps for long-term slam,” *The International Journal of Robotics Research*, vol. 28, no. 1, pp. 20–33, 2009.

[5] H. Andreasson, M. Magnusson, and A. Lilienthal, “Has somethong changed here? autonomous difference detection for security patrol robots,” in *IROS,2007. IEEE/RSJ International Conference on*. IEEE, 2007, pp. 3429–3435.

[6] J. Saarinen, T. Stoyanov, H. Andreasson, and A. J. Lilienthal, “Fast 3d mapping in highly dynamic environments using normal distributions transform occupancy maps,” in *IROS, 2013 IEEE/RSJ International Conference on*. IEEE, 2013, pp. 4694–4701.

[7] M. Gunther, T. Wiemann, S. Albrecht, and J. Hertzberg, “Building semantic object maps from sparse and noisy 3d data,” in *IROS, 2013 IEEE/RSJ International Conference on*. IEEE, 2013, pp. 2228–2233.

[8] J. Mason and B. Marthi, “An object-based semantic world model for long-term change detection and semantic querying,” in *IROS, 2012 IEEE/RSJ International Conference on*. IEEE, 2012, pp. 3851–3858.

[9] H. S. Koppula, A. Anand, T. Joachims, and A. Saxena, “Semantic labeling of 3d point clouds for indoor scenes,” in *Advances in Neural Information Processing Systems*, 2011, pp. 244–252.

[10] M. Ruhnke, B. Steder, G. Grisetti, and W. Burgard, “Unsupervised learning of 3d object models from partial views,” in *ICRA, 2009. IEEE International Conference on*. IEEE, 2009, pp. 801–806.

[11] E. Herbst, P. Henry, X. Ren, and D. Fox, “Toward object discovery and modeling via 3-d scene comparison,” in *ICRA, 2011 IEEE International Conference on*. IEEE, 2011, pp. 2623–2629.

[12] E. Herbst, X. Ren, and D. Fox, “Rgb-d object discovery via multi-scene analysis,” in *IROS, 2011 IEEE/RSJ International Conference on*. IEEE, 2011, pp. 4850–4856.

[13] P. Alimi, D. Meger, and J. J. Little, “Object persistence in 3d for home robots,” 2012.

[14] R. Finman, T. Whelan, M. Kaess, and J. J. Leonard, “Toward lifelong object segmentation from change detection in dense rgb-d maps,” in *Mobile Robots (ECMR), 2013 European Conference on*. IEEE, 2013, pp. 178–185.

[15] N. Beuter, A. Swadzba, F. Kummert, and S. Wachsmuth, “Using articulated scene models for dynamic 3d scene analysis in vista spaces,” *3D Research*, vol. 1, no. 3, pp. 1–13, 2010.

[16] T. Stoyanov, M. Magnusson, and A. J. Lilienthal, “Point set registration through minimization of the l 2 distance between 3d-ndt models,” in *ICRA, 2012 IEEE International Conference on*. IEEE, 2012, pp. 5196–5201.

[17] D. Chetverikov, D. Svirko, D. Stepanov, and P. Krsek, “The trimmed iterative closest point algorithm,” in *Pattern Recognition, 2002. Proceedings. 16th International Conference on*, vol. 3. IEEE, 2002, pp. 545–548.

[18] R. B. Rusu, G. Bradski, R. Thibaux, and J. Hsu, “Fast 3d recognition and pose using the viewpoint feature histogram,” in *IROS, 2010 IEEE/RSJ International Conference on*. IEEE, 2010, pp. 2155–2162.

[19] G. Grisetti, C. Stachniss, and W. Burgard, “Improved techniques for grid mapping with rao-blackwellized particle filters,” *Robotics, IEEE Transactions on*, vol. 23, no. 1, pp. 34–46, 2007.

[20] D. Fox, “Adapting the sample size in particle filters through kld-sampling,” *The international Journal of robotics research*, vol. 22, no. 12, pp. 985–1003, 2003.

[21] R. B. Rusu, Z. C. Marton, N. Blodow, M. Dolha, and M. Beetz, “Towards 3d point cloud based object maps for household environments,” *Robotics and Autonomous Systems*, vol. 56, no. 11, pp. 927–941, 2008.