<http://www.diva-portal.org>

Postprint

This is the accepted version of a paper presented at *2017 European Conference on Mobile Robots (ECMR)*.

Citation for the original published paper:

Welle, M., Ericson, L., Ambrus, R., Jensfelt, P. (2017)

On the use of Unmanned Aerial Vehicles for Autonomous Object Modeling.

In:

N.B. When citing this work, cite the original published paper.

Permanent link to this version:

<http://urn.kb.se/resolve?urn=urn:nbn:se:kth:diva-216994>

On the use of Unmanned Aerial Vehicles for Autonomous Object Modeling

Michael C. Welle*, Ludvig Ericson*, Rares Ambrus* and Patric Jensfelt*

Abstract—In this paper we present an end to end object modeling pipeline for an unmanned aerial vehicle (UAV). We contribute a UAV system which is able to autonomously plan a path, navigate, acquire views of an object in the environment from which a model is built. The UAV does collision checking of the path and navigates only to those areas deemed safe. The data acquired is sent to a registration system which segments out the object of interest and fuses the data. We also show a qualitative comparison of our results with previous work.

I. INTRODUCTION

Mobile robots are slowly making their way into everyday life, with robustness and operational up-time increasing every year. [1] report deploying mobile robots in unstructured environments (offices and elderly care homes) for durations of up to 6 months. Such experiments are pushing the boundaries of what these systems are capable of, and widen the frontier to the next set of issues to be addressed, such as exposure to large amounts of data, learning patterns about the environment, life-long robust localization and navigation, etc.

For robots to operate successfully for extended periods of time, their understanding of the environment needs to adapt as new data becomes available. In our work we are interested in analysing changes in the environment, and in building predictive models of where objects or people are likely to be at some future time. The basis for this is a robust perception system, able to reliably segment, model and re-identify objects of interest in the environment.

In previous work [2] we have looked at autonomously navigating around objects of interest and acquiring multiple views, which are fused into canonical models of the objects. We have seen that recognition of these objects in future observations increases with the number of views initially acquired by the robot. Unfortunately, when deploying such systems in unstructured environments (e.g. an office or a home) for extended periods of time, it quickly becomes apparent that a mobile robot's path is quite often obstructed by natural clutter (tables, chairs, etc.). This implies that in most situations a robot can potentially navigate to one or two additional vantage points within a room, which is often not enough.

In this work we address the first steps in solving this issue. Our aim is to augment the capabilities of an indoor mobile robot by pairing it with an Unmanned Aerial Vehicle (UAV). UAVs have a much wider reach as compared to wheeled

Fig. 1: The quadrotor drone used in the experiments. It is equipped with a Primesense RGBD camera looking forward and a set of reflective IR-markers for motion capture positioning.

mobile robots, and would thus allow the exploration of the environment from new angles which would otherwise be inaccessible. This would also allow the robot to overcome issues such as missing data due to occlusions or oblique surfaces, and would facilitate applications such as tracking or recognition. However, unlike mobile robot navigation, autonomous indoor flying is still very much a challenge, both technical and from a safety point of view. While a number of commercial standard mobile robot platforms are available with off-the-shelf navigation and localization capabilities, UAVs for indoor use are often built from scratch to fit the needs of the research in question. In addition, the flight time is limited to only a few minutes.

In this paper we present an end to end object modeling pipeline with an emphasis on the part from where an object to be observed has been identified. To limit the scope and focus on the perception and object modeling part, we make a number of simplifying assumptions to account for the abovementioned challenges. We make use of a motion capture system for positioning the drone. In the past we have shown that a mobile robot can robustly segment out objects of interest in the environment through change detection [3]. This is what we envision as the means with which the objects to be observed are generated. In this work we use a simpler method, which facilitates running experiments with the UAV in a safer, more confined space (a cage with nets on all sides).

*The authors are with the Centre for Autonomous System at KTH Royal Institute of Technology, SE-100 44 Stockholm, Sweden {mwelle, ludv, raambrus, patric}@csc.kth.se

The contribution of this paper is a UAV system which is able to autonomously plan and follow a path, while keeping an object in the environment in view. The UAV does collision checking of the path and navigates only to those areas deemed safe. While navigating, the UAV collects RGBD views which are sent to a registration system that segments out the object of interest and fuses the data. We also show a qualitative comparison of our results with previous work.

II. RELATED WORK

This paper presents a system and hence the related work is rather broad in scope. We focus here on discussing related work for view planning and object modeling.

In [4] the view planing problem (VPP) is surveyed. In [5] the authors propose an information entropy-based approach to the VPP. With a metric that measures the information gain for a possible next view point of a camera, they are able to pick the next best view out of the set containing all possible next view points. Repeating this in succession gives a path that gives a solution to the view planning problem. In [6] autonomous multicopters are used to build a 3-D model of outdoor structures. [7] discusses the setup of a UAV outdoors but adding an obstacle avoiding system. Similarly, [8] employs a freely available 2-D map of buildings in order to construct a rough 3-D model and inspect and refine it with a UAV. Bircher et al. [9] focus on structural inspection in their work, and they employ a two-step optimization paradigm to find good viewpoints. In the context of GPS-denied environment [10] presents an online path planning approach in cluttered environments. An area that has many similarities to the UAV research is underwater robots. They also have more degrees of freedom compared to a regular wheeled robot on land. Work presented in [11] shows a sampling based design of an inspection route. Also from the underwater domain, [12] presents a "Next Best Underwater View" that considers the placement of a light source. The approach can deal more effectively with distortions during the mapping process.

Our work also deals with object modeling by fusing RGBD data. Closest to our work is that of Faulhammer et al [2], where object models are created autonomously with a mobile robot, by acquiring additional views. The biggest difference between our work and that of [2] is that we run our experiments on a UAV - a platform much more difficult to control. In addition, while [2] has access to the robot odometry and uses a camera tracker to obtain the camera poses, our registration method uses a minimization framework which exploits the structure of our problem, and which does not rely on an initial guess for the camera poses. Since our aim and that of [2] are similar, we show a qualitative comparison with their method in the results section. The work of Prank et al [13] also deals with object model creation, however the focus is on a user-friendly system which allows the easy and robust creation of such models on a turntable. We include the models we obtain of our objects using their method in the results section.

Fig. 2: An overview of the system presented in the paper. The input is a partial object to be observed and the output is a (more) complete object model. The main steps in the project are view planning, plan execution and object modeling data fusion.

Finally, we obtain the resulting object model through a scene differencing operation. Herbst et al. [14] and Finman et al. [15] also use scene differencing for segmenting out objects, however the emphasis is not on creating accurate 3D models, but either on SLAM or on improving segmentation. Moreover, neither method explicitly addresses the view planning problem in the context of an autonomous agent.

The related works mentioned here do solve parts of our problem but to the best of the authors knowledge we present the first end to end solution for autonomous object modeling using UAVs in indoor environments including view planning with collisions avoidance.

III. OVERVIEW

An overview of our system is shown in Fig. 2. The input is a cluster of points corresponding to a partial observation of an object and the output is an object model. The main steps in the pipeline are view planning, plan execution and object modeling data fusion. Sections IV-A, IV-B and IV-C describe these in more detail. Figure 3 shows a snapshot from one of the experiments. We can see to the left the UAV hovering near an object sequentially moving between the view points generated by the view planning component. The plan is checked for collisions, in this case typically caused by the walls of the cage. In the upper right corner the camera image at the same point in time is shown, with the object clearly in view. The lower right corner show the raw RGBD frame along with the view plan.

A. Experimental setup

To better understand some of the decisions made in the method section we present our experimental setup already here.

1) *Hardware:* The UAV we use is shown in Figure 1. It is a custom made platform with a 250mm base (distance between motor axes) and a PX4 PixRacer flight controller. The flight controller is connected via a serial link to an onboard Intel i7 NUC computer where the computations are performed. The main sensing modality is a Primesense 1.08 RGBD camera. We experimented with a pan-tilt-unit initially to actuate the camera and thereby increase the degrees of freedom available for the view planing. However, due to weight limitations we had to settle for a staic mounting where the camera is tilted down an angle $\tau = 37^\circ$. The vertical fields of view of the camera is $\alpha = 45^\circ$. The total weight of the UAV including a 5000mAh battery is approximately 1.7kg. We use a an Optitrack motion capture

Fig. 3: A snapshot from one of the experiments. Left: A view of the UAV and the scene it looks at. Upper right: The current view from the camera. Lower right: The view plan and the raw data from the camera.

system. All experiments are performed inside a 3x3x3m cage with protective nets.

B. Generating the initial partial object

In [2], [3] we have shown how objects can be segmented out from a static background. We consider this output to be the input for our system. Because of the location for the experiments in this paper, a cage with mattresses on the floor, we are not able to let our wheeled robot generate this input. Instead we implemented a simple method for obtaining the input cluster. We first obtain the static background by executing a flight path with the UAV inside the cage and collecting RGBD data. Next, we place a new object in the environment. We use the UAV to acquire an RGBD snapshot of the environment. We segment the point cloud by removing all points below a certain height threshold, and we select the cluster of points closest to the camera. This simple segmentation method requires user input (i.e. the height at which to filter the point cloud). However, by placing the object on the same supporting object throughout the experiments this input is only needed once, while allowing us to remove the dependency on the mobile robot and focus on flying the drone, acquiring data and filtering it.

IV. METHOD

In this section we describe our approach to view planning and object modeling.

A. View planning

We approximate the object, which is only partially known, using a vertical cylinder. This is clearly a simplification but

experiments show that it is a good initial model. Let r and h be the radius and height of this cylinder respectively. Figure 4 illustrates the cylinder and the way we calculate the view point candidates in cylindrical coordinates. From before we have that the vertical field of view is α and the downward tilt angle of the camera is τ . We want to keep the objects in the center of the view. We furthermore want to provide some margin of error in the positioning. We define this in terms of an angle β with which the field of view is reduced at each end. In our experiments we use $\beta = 10^\circ$. We can calculate Z and R from the figure according to

$$Z = \frac{t_+(2rt_- + h)}{t_+ - t_-} \quad (1)$$

$$R = \frac{Z}{t_+} + r \quad (2)$$

where

$$t_- = \tan(\tau - (\frac{\alpha}{2} - \beta)) \quad (3)$$

$$t_+ = \tan(\tau + (\frac{\alpha}{2} - \beta)) \quad (4)$$

This defines a circle with radius R and located $Z + Z_0$ above the ground. We sample N view points on this circle evenly spread out along the circle. Given a 2-D map of the test environment, a simple collision check based on proximity is employed in order to eliminate view points that are not reachable for the UAV.

Fig. 4: View point calculation based on a cylindrical approximation of the object and trying to keep this cylinder in the center of the image with a minimum angular margin of β .

B. Plan execution

To execute the plan, the UAV checks its current position and moves autonomously to the nearest way point. The UAV reaches the way point when it is within some tolerance, T_p^1 , in X, Y and Z as well as a heading tolerance in yaw, T_a^1 . This is the starting point for the view acquisition, unless some of view points have been removed, in which case it moves from way point to way point to the closest end of the circle segment broken by obstacles. Then it starts to position itself with a tolerance of T_p^2 in X,Y,Z and T_a^2 tolerance in yaw. The UAV holds this position for up to $5sec$ before autonomously taking a snapshot of the RGBD-pointcloud. This procedure is repeated until snapshots from all reachable viewpoints are taken. The UAV continues to traverse the viewpoints back in a safe manner until it reaches its starting view-point after which it lands at the initial position. In our experiments we use $T_p^1 = 0.4m$, $T_a^1 = 50^\circ$, $T_p^2 = 0.1m$ and $T_a^2 = 10^\circ$,

C. Object modeling

Following the view planning and plan execution (data acquisition step), a number of RGBD views $V = V_i$ containing the object of interest have been collected autonomously by the UAV. In addition, we also have access to the RGBD views $R = R_i$ collected before the object was introduced in the environment as reference. We perform a number of registration steps to align the object views and the reference views, after which we segment out the object from the registered point clouds. We use image features for the registration, with the assumption that enough texture can be found in the environment or on the object such that salient features can be extracted successfully. Note that we do not use the position fix from the motion capture system as initial guess for the registration, and instead start with the identity matrix as the initial solution. The motion capture system has very high accuracy, which is an unrealistic assumption for an initial solution in a real world environment (i.e. mobile robot odometry tends to be noisier).

We first register the reference views $R = R_i$. For each image R_i , we extract SIFT images features [16], and for every pair R_i and R_j we compute feature correspondences

using the SIFT descriptors. We augment each remaining feature with the depth value from the corresponding RGBD frame, and we perform a RANSAC [17] step to remove spatially inconsistent matches. The remaining matches are of the form (P_i, P_j) with $P_i \in R_i$ and $P_j \in R_j$, $P_i = (X, Y, Z)$ - coordinates in the camera frame of reference. Registering these views in a common frame of reference is done using image features in a least squares minimization framework. We define the transformations T_{R_i} and we solve the following minimization problem:

$$\min_{T_{R_i}, T_{R_j}} \sum_i \sum_j \|T_{R_i} \cdot P_i - T_{R_j} \cdot P_j\|^2 \quad (5)$$

Fig. 5: Object modeling results: a, b and c) show the registered frames V_i for three experiments.

Note that when acquiring the reference views R , the UAV is free to collect data without aiming at a specific object, which ensures a uniform and complete coverage of the

Fig. 6: Qualitative results of the object models built: (a) Ground truth models built on a turntable using [13]; (b) Results from the controlled experiments of [2]; (c) Uncontrolled experiments of [2]; and (d) Our object models.

environment. This coverage ensures that enough features and enough overlap is present between frames for a successful registration.

The next step is to register the object views $V = V_i$. This is more challenging, as the drone’s path can be obstructed by obstacles such as walls, resulting in fewer frames and less information for the registration. Moreover, depending on the object introduced in the scene, the depth information can sometimes be unreliable, especially if the camera view axis incidental to the surface of the object is too oblique. As above, we are interested in finding the transformations T_{V_i} which bring the views $V = V_i$ into a common frame of reference. From our experiments, solving Eq. 5 for the set of views $V = V_i$, where less information is available sometimes results in failure. Instead we use the set of views R_i as reference, and keep the transforms computed earlier T_{R_i} fixed.

As before, we compute SIFT feature correspondences between image pairs V_i and V_j , augmented with depth information. Next, we compute SIFT feature correspondences between image pairs V_k and R_l . The minimization problem becomes:

$$\begin{aligned} \min_{T_{V_i}, T_{V_j}} \sum_i \sum_j \|T_{V_i} \cdot P_i - T_{V_j} \cdot P_j\|^2 + \\ + \sum_k \sum_l \|T_{V_k} \cdot P_k - T_{R_l} \cdot P_l\|^2 \end{aligned} \quad (6)$$

In Eq. 6, we force the views V_i to register with each other and against the reference views R_i . Keeping the transformations T_{R_i} fixed is required, otherwise the registration of R_i might diverge if the quality of data in V_i is too poor.

We perform a final registration step where we allow only small changes to the transforms T_{V_i} and T_{R_i} . This step fine-tunes the registration, and has the same shape as 6, except both T_{V_i} and T_{R_i} are varied by the optimizer. The results of the registration can be seen in Fig. 5.

For all registration steps we use the Ceres optimization engine [18], and the transforms (unless previously computed) are initialized with the identity matrix. Note that because we estimate transforms between all pairs of frames, the notion of “loop-closure” is implicitly taken into account in our pipeline.

Finally, we segment out the object model by taking a point cloud difference between the view point clouds V_i and the reference point cloud R_i . The results of the segmentation are shown in Fig. 6. We also perform a voxel-grid downsampling operation of the resulting object point cloud, thus removing some of the clutter and noise.

V. RESULTS

We perform experiments using five household items. Before each run, we first perform a flight pass with the UAV and stop at specified way points in the cage. This gives us the map we compare against, which would normally be provided by the mobile robot. We then place one of the objects in the

environment (in various positions, including corners), and we run the segmentation procedure described in Sec. III-B. Further, we collect data at the viewpoints computed and we extract one object model for each experiment.¹

We compare our results qualitatively with the results obtained by [2], where a mobile robot was used to acquire views and build models of the objects autonomously. Two types of experiments were performed in [2]: *controlled* and *uncontrolled*. In the controlled experiments, the object was placed in an accessible area, such that the robot was able to navigate all around and collect additional views from all sides. In the uncontrolled experiments, the object was placed in more natural but less accessible locations, and the robot was only able to collect a few additional views. Finally, we also show ground truth point cloud models of the objects built using a turntable with the method described in [13].

The results are shown in Fig. 6. We notice that the ground truth models are the sharpest, with the least amount of blur or noise, and with the textures clearly visible. Fig. 6 b) and c) show the progression when the objects are placed from more to less accessible locations. The degradation in quality can readily be observed when the robot can only take a few snapshots of the objects from further away. In contrast, our results in Fig. 6 d) are closer in quality to the ones of [2] in the controlled experiments, even though we conduct our experiments on a platform which is much more difficult to control. We note that our registration and modeling step performs successfully, and that the resulting model contains much more information than a single scan of the object would.

As the drone is not limited to the floor for navigation, we conclude that the pipeline proposed has the capability of augmenting a mobile robot system by safely navigating to previously inaccessible locations, collecting data reliably and using the data to create object models autonomously.

VI. CONCLUSION

In this paper we presented an end to end object modeling pipeline for an unmanned aerial vehicle (UAV). We explained how to generate a view plan, execute it to acquire data and how to use this data to build object models. Our experiments show that the quality of our models are close to those generated by a mobile robot in situations where the objects have been placed in easy to access locations as presented in current state of the art in object modeling [2]. To the best of our knowledge this is the first end to end solution for autonomous object modeling using a drone in indoor environments including view planning with collision avoidance.

In future work we want to relax the assumption of having positioning information from the motion capture system and we want to form the team between the wheeled mobile robot and the UAV which was the motivation for the work in the first place. We also want to investigate replacing the wheeled mobile robot all together with one or several drones performing modeling of indoor spaces.

ACKNOWLEDGMENT

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] N. Hawes, C. Burbridge, F. Jovan, L. Kunze, B. Lacerda, L. Mudrová, J. Young, J. Wyatt, D. Hebesberger, T. Körner, *et al.*, “The strands project: Long-term autonomy in everyday environments,” *arXiv pre-print arXiv:1604.04384*, 2016.
- [2] T. Fäulhammer, R. Ambruş, C. Burbridge, M. Zillich, J. Folkesson, N. Hawes, P. Jensfelt, and M. Vincze, “Autonomous learning of object models on a mobile robot,” *IEEE Robotics and Automation Letters*, vol. 2, no. 1, pp. 26–33, 2016.
- [3] R. Ambruş, N. Bore, J. Folkesson, and P. Jensfelt, “Meta-rooms: Building and maintaining long term spatial models in a dynamic world,” in *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pp. 1854–1861, IEEE, 2014.
- [4] W. R. Scott and G. Roth, “View Planning for Automated Three-Dimensional Object Reconstruction and Inspection,” vol. 35, no. 1, pp. 64–96, 2003.
- [5] Y. F. Li and Z. G. Liu, “Information entropy-based viewpoint planning for 3-D object reconstruction,” *IEEE Transactions on Robotics*, vol. 21, no. 3, pp. 324–337, 2005.
- [6] K. Schmid, H. Hirschmüller, A. Domel, I. Grixia, M. Suppa, and G. Hirzinger, “View planning for multi-view stereo 3D Reconstruction using an autonomous multicopter,” *Journal of Intelligent and Robotic Systems: Theory and Applications*, vol. 65, no. 1-4, pp. 309–323, 2012.
- [7] X. Z. Peng, H. Y. Lin, and J. M. Dai, “Path planning and obstacle avoidance for vision guided quadrotor UAV navigation,” *IEEE International Conference on Control and Automation, ICCA*, vol. 2016-July, pp. 984–989, 2016.
- [8] W. Jing, J. Polden, P. Y. Tao, W. Lin, and K. Shimada, “View Planning for 3D Shape Reconstruction of Buildings with Unmanned Aerial Vehicles,” vol. 2016, no. November, pp. 13–15, 2016.
- [9] A. Bircher, K. Alexis, M. Burri, P. Oettershagen, S. Omari, T. Mantel, and R. Siegwart, “Structural Inspection Path Planning via Iterative Viewpoint Resampling with Application to Aerial Robotics,”
- [10] S. Lai, K. Wang, H. Qin, J. Q. Cui, and B. M. Chen, “A robust online path planning approach in cluttered environments for micro rotorcraft drones,” *Control Theory and Technology*, vol. 14, no. 1, pp. 83–96, 2016.
- [11] B. Englot and F. S. Hover, “Three-dimensional coverage planning for an underwater inspection robot,” *The International Journal of Robotics Research*, vol. 32, no. 9-10, pp. 1048–1073, 2013.
- [12] M. Sheinin and Y. Y. Schechner, “The Next Best Underwater View,” p. 32000, 2015.
- [13] J. Prankl, A. Aldoma, A. Svejda, and M. Vincze, “Rgb-d object modelling for object recognition and tracking,” in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pp. 96–103, IEEE, 2015.
- [14] E. Herbst, P. Henry, X. Ren, and D. Fox, “Toward object discovery and modeling via 3-d scene comparison,” in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pp. 2623–2629, IEEE, 2011.
- [15] 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*, pp. 178–185, IEEE, 2013.
- [16] C. Wu, “Siftgpu: A gpu implementation of scale invariant feature transform (sift),” 2007.
- [17] M. A. Fischler and R. C. Bolles, “Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography,” *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.
- [18] S. Agarwal, K. Mierle, *et al.*, “Ceres solver,” 2013.

¹The data can be found online at goo.gl/gLn8D2