Robust Localization of Occluded Targets in Aerial Manipulation via Range-Only Mapping

Xiaoyu Zhang, Yin Zhang, Peidong Liu, and Shiyu Zhao

***Abstract*—This paper studies the problem of target localization in aerial manipulation tasks. When an aerial robot flies close to a target to manipulate, the target would be occluded by the onboard robotic manipulator occasionally or for a long period of time. It is, however, necessary to continuously localize the target even when it is occluded. This is a unique problem in aerial manipulation and has not been addressed in other tasks such as ground manipulation. To solve this problem, we propose a new approach of target-centered range-only mapping based on RGB-D measurements. Different from the traditional map representation where the 3D locations of the feature points are stored in a global reference coordinate frame, we store the distances from the target to the environmental feature points only. It thus results in a coordinate-free target-centered range-only map. When the target is occluded, it can still be localized accurately based on the map. The proposed approach is verified by both synthetic and real datasets. The experimental results demonstrate the superior performance of our proposed approach to the state-of-the-art RGB-D ORB-SLAM3 method.**

***Index Terms*—Aerial Systems: Applications; Localization**

I. Introduction

AERIAL robotic manipulator is a new type of robot that combines micro aerial vehicles (MAVs) with onboard robotic arms and/or specialized devices [1]–[3]. It thus takes advantage of the rapid movement capability of MAVs and the precise operating capability of robotic manipulators. Aerial manipulation has great potential for real-world applications and receives increasing attention in recent years [3]. However, current research mainly focuses on the platform design [4], [5] and control systems [6], [7] of aerial manipulation. Target sensing in aerial manipulation has not received sufficient attention yet (a review of related work is given in Section II).

The specific target sensing problem addressed in this paper is how to continuously localize a target even when it is occluded. In particular, the location of the target relative to the aerial manipulator must be continuously estimated to achieve accurate target manipulation. In some work, the camera is mounted on the end effector to avoid occlusion caused by the manipulator [8], [9]. For such hand-in-eye configuration, however, the camera would lose sight of the target easily when it is close to the target, since tiny movement of the MAV (e.g., caused by air turbulence) would cause large deviation on the camera position [8]. Besides, due to the limited space and payload of the aerial manipulator, the camera often cannot be mounted on the manipulator but is mounted on the MAV (e.g., the MAV shown in Fig. 1). Thus, the manipulator can easily occlude the target. Such occlusion is hardly avoidable in many scenarios [10], [11]. Therefore, to ensure successful manipulation, the target needs to be reliably and continuously localized even when it is occluded for a long period of time.

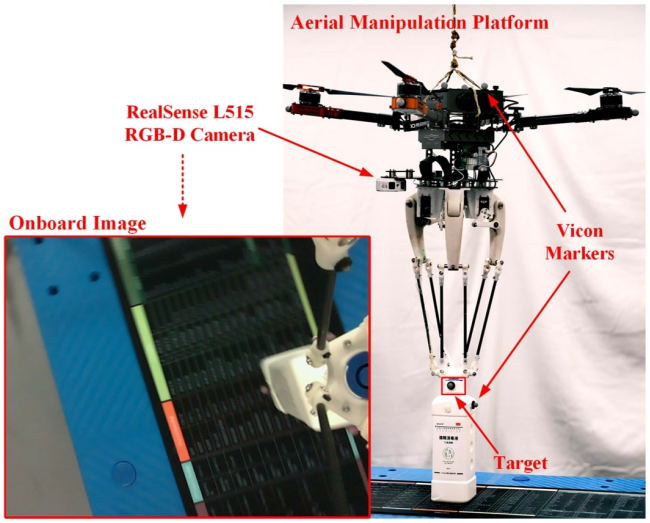

Fig. 1: The aerial manipulation platform used in our work. The RGB-D camera is downward-facing. When the robotic arm is sufficiently close to the target (i.e., the bottle cap), the target is completely occluded as shown in the onboard image in the bottom-left corner.

This is a unique problem in aerial manipulation tasks and has not been specifically addressed in other tasks such as ground manipulation. That is because this problem could be well avoided in ground manipulation by some methods, for example, relocating cameras to appropriate locations [12] or even using multiple cameras so that at least one camera could see the target while the others are occluded [13], [14]. These methods are feasible for ground manipulation tasks where the constraints of the sensor placement are not strict. They are, however, inapplicable to aerial manipulation tasks due to rigid sensor placement constraints.

Although it has not been specifically addressed in the literature, localizing occluded targets may be potentially solved by simultaneous localization and mapping (SLAM), which aims to estimate the camera pose and build the environmental map simultaneously [15]. In particular, when the camera is

Manuscript received: September, 9, 2021; December, 8, 2021; January, 10, 2022.

This paper was recommended for publication by Editor Pauline Pounds upon evaluation of the Associate Editor and Reviewers' comments.

The authors are with the School of Engineering at Westlake University and the Institute of Advanced Technology at the Westlake Institute for Advanced Study, Hangzhou, China. X. Zhang completed this work when he was a Research Assistant at Westlake University and is currently with the Department of Mechanical and Automation Engineering at the Chinese University of Hong Kong, China. Corresponding author: zhaoshiyu@westlake.edu.cn

Digital Object Identifier (DOI): see top of this page.

not able to see the target, it could still estimate its pose relative to the map based on other seeable features. While the target's location in the map is already estimated, the relative location between the target and the camera could be recovered indirectly. However, this approach relies on the estimated pose of the camera, which may be corrupted by errors. The estimation error in the camera pose will be propagated into the target localization error. It is meaningful to study a method that does not rely on the camera pose.

This paper proposes a novel approach to reliably and accurately localize targets that may be occluded for long periods of time. In particular, a novel concept of target-centered range-only map is proposed. The range-only map only stores the distances between the target and the environmental feature points. When the target can be detected by the camera, its distances to other feature points are stored in the map. When the target is occluded, the feature points detected in the present image frame could be matched to the features in the map, and the relative location between the target and the camera can be recovered based on the prior stored distances and the location of the matched feature points in the current camera coordinate frame.

The novelty of our approach is that it is coordinate-free because distances can be defined without coordinate frames. Thus, the target localization accuracy will not be downgraded by the drift of the camera pose. Moreover, although the proposed approach is motivated by the case where the target is occluded in aerial manipulation, it could also tackle the case where the target is not found in the image frame due to other reasons such as the target is out of the field of view of the camera. Even if the target is visible to the camera, it is still possible that the target could not be detected correctly due to, for example, illumination or appearance changing. The proposed approach could handle these cases as well. Our experiments on both synthetic and real datasets indicate that the proposed approach outperforms the state-of-the-art RGB-D ORB-SLAM3 approach in terms of localization accuracy. Our approach is also computationally efficient and could be run in real-time at 24 frames per second. An associated video is provided online [1].

II. Related Work

A. Target localization in aerial manipulation

Aerial manipulation is a relatively new research area. While the existing studies mainly focus on control and platform systems, sensing problems in aerial manipulation have not received sufficient attention yet.

Most existing target localization methods in aerial manipulation rely on either artificial markers or external motion capturing systems. For example, in the work of [16], AprilTags are placed near the target area in advance so that the aerial robot could always see the AprilTags while performing manipulation. Since the relative pose between the AprilTags and the target is calibrated in advance, by estimating the relative pose with respect to the AprilTags, the aerial robot could estimate the relative pose to the target indirectly. In [17], an external motion capturing system (i.e., VICON) is used to accurately localize the MAV with respect to the target. Zhao et al. [11] design a target recognition and localization vision system for an unmanned helicopter to autonomously grasp a bucket. However, the vision system could not see the bucket anymore when the helicopter is sufficiently close to the bucket due to the offset of the griper and the camera. Laiacker et al. [10] install a 7-DoF KUKA robotic arm on a helicopter for aerial manipulation. When grasping the target, the big robotic arm causes severe occlusion of the target. To address this problem, the authors place multiple artificial markers around the target. When the target is occluded, the surrounding markers can still be used to localize the target.

In summary, the problem of target occlusion has already been encountered in some existing studies. However, most works simply avoid this problem by introducing, for example, artificial markers whose relative poses to the target are calibrated in advance. As a comparison, our approach is free of this requirement.

B. Target occlusion in traditional robotic arm manipulation

Camera is also widely used for target localization in traditional robotic arm manipulation. However, different from aerial manipulation, traditional robotic manipulation has many ways to avoid target occlusion.

One trivial solution is to place multiple cameras in the environment (i.e., not on the robotic arm itself) to avoid target occlusion [13], [14]. This solution is, however, adequate only for well-controlled working environments. In [18], [19], the possible positions causing target occlusion are predicted at the algorithm level and are avoided during path planning. Similarly, [20] formulates occlusion avoidance as one of the optimization goals for its automatic control algorithm. These methods are not applicable in aerial manipulation because there are no sufficient degrees of freedom for optimization. In recent years, several neural network-based methods [21] are used to estimate the pose of the target and can deal with target occlusion to a certain extent. These methods, however, can only deal with objects of known models and still need to see parts of the target.

As a comparison, our approach does not have this limitation or requirement of considering occlusion avoidance in the control system design and would thus be applicable to broader kinds of operating scenarios.

III. Proposed Approach

An overview of the proposed approach is illustrated in Fig. 2 and briefed as follows. The approach takes RGB-D images as input and then extracts Oriented FAST and Rotated BRIEF (ORB) feature points [22]. The reason that ORB features are used is that they are fast to extract and match, and also robust to illumination and viewing angle changes. In the current pipeline, the target is manually labelled in the first frame and then represented by an ORB feature point, which is known as the target point. If the target point is matched, the target is supposed to be detected in the image; if not, the

[1] https://youtu.be/t6-0zaRuFIY

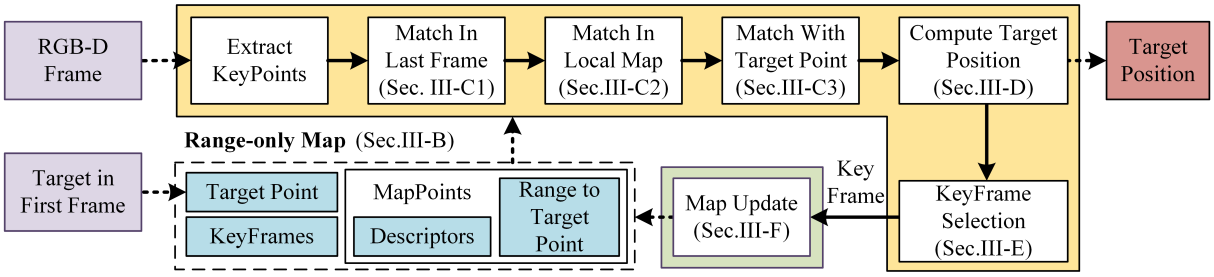

Fig. 2: Overview of the proposed range-only mapping approach. The white rectangular boxes refer to the algorithmic modules in which the corresponding subsection is labelled, and other colored rectangular boxes refer to the data. The yellow region illustrates the processing of every frame. The range-only map is constructed from the keyframes only.

target is supposed to be occluded or out of the field of view. More sophisticated automatic target detection methods could be easily integrated into our pipeline according to different tasks in the future.

Based on the depth measurements, the relative distances between the target point and the environmental feature points can be easily computed. Thus, a target-centered range-only map is constructed. To further improve the robustness, the measurements from multiple views are integrated to update the map. During target localization, the detected environmental features in the current frame are matched against that from the map. Based on the range-only map and the depth of the matched points, the approach can recover the relative location between the target and the current camera, even when the target is not detectable due to occlusion, out of the field of view, etc. Note that the approach does not rely on the camera pose. Therefore, potential camera pose drift will not downgrade the accuracy of target localization.

The details of each component of the pipeline are given below.

A. Notations

We denote the current processing frame as $\mathcal{F}^i$, and thus $\mathcal{F}^{i-1}$ is the last processed frame. To reduce the amount of data, keyframes $\mathcal{F}^j_{key}$ are subsampled from all frames to be stored. The proposed approach maintains a global range-only map $\mathcal{M}$, a part of which is selected for fast feature matching in frame $\mathcal{F}^i$ and denoted as local map $\mathcal{M}^i_{local}$.

B. Range-only map representation

The range-only map $\mathcal{M}$ stores the distances between the target point and environmental feature points. It mainly consists of map points as well as their associated keyframes.

Given a new keyframe $\mathcal{F}^j_{key}$, the detected feature points are back-projected into the current camera coordinate frame from the measured depth image. If the target point is matched in keyframe $\mathcal{F}^j_{key}$, its spatial distances to newly detected feature points are easily computed to update the map. The computed distance is called *directly measured distance* (DMD), as illustrated by yellow lines in Fig. 5. If the target point is not matched, the proposed approach can recover the location of the target in the current camera coordinate frame. The distance can then be further computed to update the map $\mathcal{M}$. The computed distance is known as *indirectly computed distance* (ICD), as illustrated by red lines in Fig. 5.

C. Feature point matching

To estimate the relative location between the target and the camera, the approach needs to build the correspondences between the features detected in frame $\mathcal{F}^i$ and the features in the range-only map $\mathcal{M}$. To improve both the efficiency and efficacy, the matching process is mainly divided into three steps: 1) matching against the feature points from the last frame; 2) matching the remaining unmatched feature points against that from the local map; 3) matching against the target point.

1) Match in Last Frame: Since the camera's viewpoint does not change much between two consecutive frames, it would be effective to match the newly detected feature points against that of the last frame. To improve the efficiency, each feature point from the frame $\mathcal{F}^{i-1}$ is matched against all feature points in frame $\mathcal{F}^i$ within a bounding box. In particular, the feature point $\mathbf{u}^{i-1} = (x, y)$ from frame $\mathcal{F}^{i-1}$ is matched against all feature points within a bounding box, which has side length 10 pixels and centers at $\mathbf{u}^i = (x, y)$ in frame $\mathcal{F}^i$. If the target localization is not successful in frame $\mathcal{F}^{i-1}$, the newly detected feature points are matched against all feature points from the latest previous successful frame based on bag-of-words [23].

2) Match in Local Map: For the remaining unmatched feature points from frame $\mathcal{F}^i$, the approach matches them against that from the local map $\mathcal{M}^i_{local}$, which refers to a set of keyframes that have certain common matched points with $\mathcal{F}^i$. The selection criteria are as follows:

1) The keyframes that have common matched map points with the current frame are found and the number of common matched points for each keyframe is also computed.
2) The keyframes whose number of common matched points is larger than the threshold is inserted into the local map.

More details of selecting the local map is illustrated in Algorithm 1. The number of points to be matched becomes much larger, therefore, bag-of-words is used to accelerate the feature matching in the local map.

3) Match with Target Point: The target point stored in the map $\mathcal{M}$ is also matched against that from frame $\mathcal{F}^i$. Since there is only one target feature point involved, the approach applies a brute-force matching scheme to find the correspondence. It might happen that no correct match would

Algorithm 1 Select local map $\mathcal{M}^i_{local}$ for frame $\mathcal{F}^i$

Input: $\mathcal{P}^i$: list of feature points in $\mathcal{F}^i$;
Output: $\mathcal{M}^i_{local}$;

```
Initialize Dict, M^i_local;
N_max ← 0;
for p_j in P^i do
    MP ← GetMatchedMapPoint(p_j);
    if MP ≠ 0 then
        KF ← GetMatchedKeyFrames(MP);
        for KF_m in KF do
            Dict[KF_m] ← Dict[KF_m] + 1;
            if Dict[KF_m] > N_max then
                N_max ← Dict[KF_m];
            end if
        end for
    end if
end for
for {KF_n, n} in Dict do
    if n > N_max/4 then
        M^i_local ← GetAllMapPoints(KF_n);
    end if
end for
```

be found, which is probably caused by occlusions. Then the approach recovers its relative location to ensure the success of the aerial manipulation task.

D. Target localization algorithm

Given the matched feature points and the depth image of frame $\mathcal{F}^i$, their 3D locations in the current camera coordinate frame are recovered via

$$\mathbf{p}^i_k = \lambda_k \mathbf{K} \bar{\mathbf{u}}^i_k, \tag{1}$$

where $\mathbf{p}^i_k$ is the 3D location of the matched feature point in frame $\mathcal{F}^i$, λ_k is the measured depth of the feature point, $\mathbf{K}$ is the camera intrinsic matrix, $\bar{\mathbf{u}}^i_k$ is the homogeneous representation of the pixel location of the feature point in frame $\mathcal{F}^i$.

Given the range-only map $\mathcal{M}$, the relationship between the target point and the feature points is as follows:

$$\left\| \mathbf{t}^i - \mathbf{p}^i_k \right\|_2 = d_k, \tag{2}$$

where $\mathbf{t}^i$ is the target location in the current coordinate frame of $\mathcal{F}^i$, d_k is the distance retrieved from the range-only map for the feature point.

Every matched feature point provides such an equation. In theory, a minimal set of four points that are not on the same plane can determine the unique solution of $\mathbf{t}^i$. If there are more than four points, an over-determined system of equations is built:

$$\begin{cases} \left\| \mathbf{t}^i - \mathbf{p}^i_1 \right\|_2 = d_1 \\ \left\| \mathbf{t}^i - \mathbf{p}^i_2 \right\|_2 = d_2 \\ \vdots \\ \left\| \mathbf{t}^i - \mathbf{p}^i_n \right\|_2 = d_n \end{cases} \tag{3}$$

Squaring both sides of each equation and subtracting the first equation gives a linear system of equations

$$\mathbf{A}\mathbf{t}^i = \mathbf{b}, \tag{4}$$

where

$$\mathbf{A} = \begin{bmatrix} {\mathbf{p}^i_2}^\top - {\mathbf{p}^i_1}^\top \\ {\mathbf{p}^i_3}^\top - {\mathbf{p}^i_1}^\top \\ \vdots \\ {\mathbf{p}^i_n}^\top - {\mathbf{p}^i_1}^\top \end{bmatrix} \tag{5}$$

$$\mathbf{b} = \begin{bmatrix} b_2 & b_3 & \cdots & b_n \end{bmatrix}^\top \tag{6}$$

$$b_k = \frac{{d_1}^2 - {d_k}^2 - {p^i_{1x}}^2 + {p^i_{kx}}^2 - {p^i_{1y}}^2 + {p^i_{ky}}^2 - {p^i_{1z}}^2 + {p^i_{kz}}^2}{2} \tag{7}$$

The least-squares solution of $\mathbf{t}^i$ can be computed by $\mathbf{t}^i = (\mathbf{A}^\top \mathbf{A})^{-1} \mathbf{A}^\top \mathbf{b}$. However, the least-squares approach is prone to be inaccurate due to mismatched points.

The proposed approach uses iterative optimization to compute the target point location instead. The robust kernel function used in optimization can be applied to reduce the influence of outliers. Mismatched points can also be easily filtered out. Therefore, the problem is converted to a least-squares optimization problem, and the error residual is defined as follows:

$$r^i_k = \left\| \mathbf{t}^i - \mathbf{p}^i_k \right\|^2_2 - d^2_k \tag{8}$$

The target location can then be solved by minimizing the following cost function iteratively:

$$\mathbf{t}^{i^*} = \arg\min_{\mathbf{t}^i} \sum_{k=1}^{n} \frac{1}{2} \omega_k \cdot \left(r^i_k \right)^2, \tag{9}$$

where ω_k is a weighting factor. In particular, if d_k is DMD, a larger value (e.g., 1.0) is set for ω_k; if d_k is ICD, a smaller value (e.g., 0.1) is set for ω_k. For every matched point, DMD is used if it is available, otherwise, ICD is used. The Levenberg-Marquardt method implemented in g2o [24] is used to solve the optimization problem.

The target location in frame $\mathcal{F}^{i-1}$ is used as the initial value for optimization in frame $\mathcal{F}^i$. If the target localization is not successful in frame $\mathcal{F}^{i-1}$, we randomly sample at most 50 matched points and calculate the initial value by the above least-squares approach.

To reduce the impact of outliers, the Huber kernel function [25] is used. The points with large errors are also filtered out to avoid the influence of these outliers during the optimization. If a map point has large errors in the optimization for several frames, it will be deleted from the map $\mathcal{M}$.

E. Keyframe selection

For every frame in which the target is successfully localized, the approach checks whether it meets the keyframe selection criteria. The criterion is that the matching success rate of the feature points in frame $\mathcal{F}^i$ is lower than a pre-defined threshold N_{kf}.

F. Range-only map update

The range-only map is updated from each new keyframe $\mathcal{F}^j_{key}$. If the target point is matched in keyframe $\mathcal{F}^j_{key}$, DMDs are computed from the 3D coordinates of the feature points and the target point. If the target point is not matched, ICDs are computed from the 3D coordinates of the feature points and the target location computed by our proposed approach. For matched feature points in keyframe $\mathcal{F}^j_{key}$, the average values are computed to update the corresponding feature descriptors and DMDs / ICDs of each map point. For unmatched feature points, new map points are created and their feature descriptors and DMDs / ICDs are also saved.

IV. Experimental Evaluations

The proposed approach is evaluated with both synthetic datasets and real-world experiments. All experiments run on an i7-7700HQ 2.80 Hz CPU. The location error is used to evaluate the accuracy of our approach. The efficiency of the approach is also evaluated via its total run-time.

A. Experimental results on the ICL-NUIM Dataset

ICL-NUIM [26] is a synthetical dataset, which is often used for evaluating SLAM and 3D reconstruction algorithms. The images are captured within synthetically generated indoor environments. The dataset provides images with and without noise, as well as the groundtruth of the camera poses. The data includes both RGB and depth images captured at 30 Hz, with a resolution of 640×480 pixels. In our experiments, a maximum of 1000 feature points is extracted from each frame. The target is set using a bounding box in the first frame. As the camera moves, the target will move out of the field of view of the camera. By using the images without noise and the groundtruth of camera poses, the groundtruth of the relative target location in each frame can then be computed.

TABLE I: Target localization error in the ICL-NUIM dataset. E_m denotes the error of the frames in which the target point is matched; E_u denotes the error of the frames in which the target point is not matched. The smaller average errors are labelled in bold. The number of frames for target localization is indicated under the corresponding error.

	ORB-SLAM3 (RGB-D)		Our approach	
	E_m (cm)	E_u (cm)	E_m (cm)	E_u (cm)
off_0	4.47 ± 1.6 (53)	7.67 ± 0.7 (147)	$\mathbf{2.81 \pm 0.4}$ (89)	$\mathbf{2.65 \pm 0.3}$ (111)
off_1	2.93 ± 1.0 (104)	6.08 ± 3.6 (96)	$\mathbf{2.44 \pm 0.7}$ (29)	$\mathbf{2.36 \pm 0.8}$ (171)
off_2	3.75 ± 2.4 (82)	4.39 ± 2.2 (118)	$\mathbf{2.23 \pm 0.8}$ (56)	$\mathbf{3.14 \pm 1.2}$ (144)
off_3	3.32 ± 1.1 (83)	3.59 ± 1.3 (37)	$\mathbf{3.11 \pm 2.6}$ (18)	$\mathbf{3.08 \pm 1.7}$ (102)
liv_0	6.06 ± 2.5 (223)	6.34 ± 3.0 (82)	$\mathbf{2.13 \pm 1.2}$ (180)	$\mathbf{2.45 \pm 1.0}$ (125)
liv_1	$\mathbf{1.12 \pm 0.2}$ (39)	2.85 ± 0.9 (161)	1.41 ± 0.6 (138)	$\mathbf{2.47 \pm 1.0}$ (62)
liv_2	$\mathbf{1.59 \pm 0.4}$ (33)	$\mathbf{2.36 \pm 0.8}$ (167)	1.91 ± 0.6 (45)	2.44 ± 0.7 (155)
liv_3	4.52 ± 2.1 (128)	6.47 ± 1.4 (72)	$\mathbf{1.88 \pm 0.6}$ (162)	$\mathbf{2.00 \pm 0.5}$ (38)

Fig. 3: Images from the sequence living_room_1. The frame ID is labelled in the images. The target is set by the red square in the first frame. The target moves out of the image after the 150th frame.

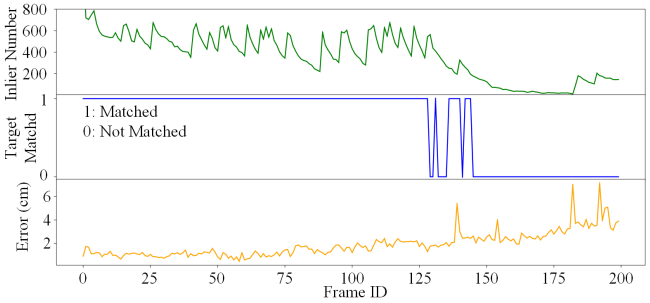

Fig. 4: Experimental results in the sequence living_room_1. The frame rate is 30 fps. The green curve indicates the number of matched points to compute the target location. The blue curve indicates whether the target point is matched in the frame, 1 denotes matched, 0 denotes not matched. The orange curve indicates the target localization error.

1) **Comparison with SLAM:** As mentioned before, SLAM is a straightforward approach to deal with the target occlusion. We compare our approach with the state-of-the-art RGB-D ORB-SLAM3 [27] system. The target is also set in the first frame for SLAM, and a global map containing the location of 3D points is constructed in SLAM. The camera pose with respect to the map is estimated and thus the relative location between the target point and the camera can be computed.

Table I gives the average target localization error and the standard deviation of the error before and after the target moves out of the image. Our approach achieves better localization accuracy in most sequences, especially for the frames that cannot match with the target point, compared to SLAM-based method. One of the reasons why SLAM-based method does not work well is that it depends on the camera pose. In SLAM, however, the estimation of both the camera pose and the target point would drift inevitably. In contrast, our approach does not depend on the camera pose and achieves higher accuracy.

2) **Localization error analysis of our approach:** Indicated by Table I, the target localization errors of our approach only change a little after the target point cannot be matched in most sequences and the errors even become smaller in some sequences (e.g., office_room_0). Even if the target point is matched in the frame, its position is also computed using the method in Sec. III-D to improve the robustness. Therefore, the estimation may be affected by some outliers. If these outliers are removed, the estimation would be more accurate, although the target point cannot be matched.

In some sequences, the target localization errors of our approach become larger after the target point cannot be matched (i.e., E_u is larger than E_m for some sequences in Table I). Fig. 3 presents the images from the sequence living_room_1.

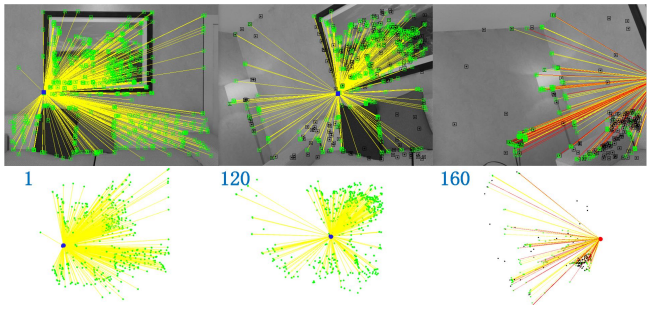

Fig. 5: Range-only map in the sequence living_room_1. The matched map points are drawn in green and unmatched ones are in black. Yellow lines indicate the DMD, and red lines indicate the ICD. The blue point is the matched target point; the red point is the localization result when the target point is not matched.

Fig. 4 indicates the target localization error, whether the target point is matched in the frame, and the number of matched points to compute the target location. It can be seen that as the camera moves, the target gradually moves out of the image after the 150th frame. As indicated in Fig. 3, the scene changes greatly and the texture in images becomes poor, therefore, the number of matched points decreases quickly. Thus, the target localization error becomes a little larger after the target point cannot be matched. The results indicate the number of matched features could affect the estimation accuracy in our approach.

Fig. 5 indicates the range-only map in some frames of the sequence living_room_1, drawn in both images and 3D space. It indicates the target gradually moves out of the image, but the target location can always be computed from environmental feature points. After the target moves out of the image, the ICDs are also involved in the target localization.

3) **Long-time target localization:** In the previous experiments, the target is localized until the camera moves to a new scene. In this long-time experiment, however, we perform target localization continuously even when the camera moves to totally different scenes and is far from the target. The target can also be localized by using the measurements of ICD, as shown in Fig. 3 and Fig. 6. The localization error curve is shown in Fig. 7. The scene of the 400th frame is totally different from the initial scene containing the target. But the relative location of the target can still be computed. Because only ICD can be obtained, the target localization error becomes a little larger. But the approach still provides a rough location of the target.

Fig. 6: Images from the sequence living_room_1. The frame ID is labelled in the images. As the camera moves, the scene is totally different from that containing the target.

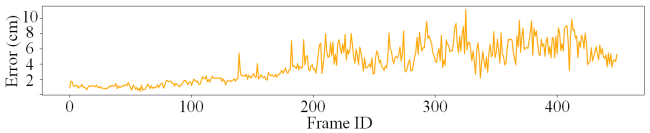

Fig. 7: Target localization error of long-time experiment in the sequence living_room_1. The frame rate is 30 fps.

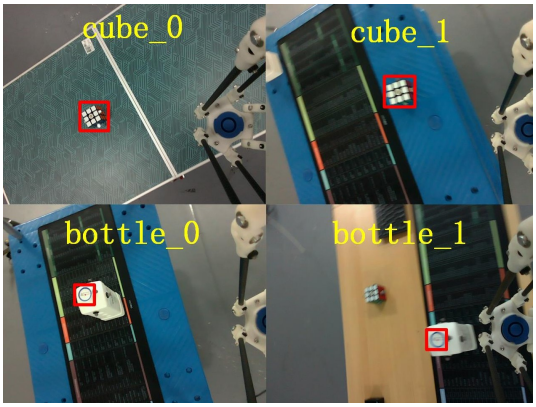

Fig. 8: Real-world experiments. A bottle and a Rubik's cube are used as the targets, testing in different scenes.

B. Real-world experiments

To further verify our approach in practical applications, we use an aerial robot equipped with a delta arm to collect data and perform the target localization experiments. The platform is shown in Fig. 1. In the experiments, a downward-facing RealSense L515 RGB-D camera is used. The image resolution is 640×480 pixels, the frame rate is about 20 Hz, and a maximum of 1000 feature points is extracted from each frame. An indoor motion capturing system (i.e., VICON) is used to provide the groundtruth of the target location and the MAV pose. The relative pose between the camera and the MAV is calibrated using the hand-eye calibration tool [28]. Since different features my be extracted as target points in different sequences, we mainly focus on comparing the localization error before and after the target is occluded to prove the effectiveness of our approach.

As shown in Fig. 8, we use a bottle and a Rubik's cube as the targets and perform the experiments in different scenes. In the experiments, the robotic arm is also captured in the image. To remove the features from the robotic arm, a minimum effective depth value (0.4 m) is set. Such removal could be achieved by other advanced methods (e.g., robotic arm segmentation).

TABLE II: Target localization error in real-world experiments. E_m denotes the error of the frames in which the target point is matched; E_u denotes the error of the frames in which the target point is not matched. The smaller average errors are labelled in bold. The number of frames for target localization is indicated under the corresponding error.

	ORB-SLAM3 (RGB-D)		Our approach	
	E_m (cm)	E_u (cm)	E_m (cm)	E_u (cm)
cube_0	1.94 ± 0.2 (114)	2.19 ± 0.2 (129)	$\mathbf{1.22 \pm 0.2}$ (128)	$\mathbf{1.32 \pm 0.2}$ (113)
cube_1	1.37 ± 0.2 (3)	2.59 ± 1.0 (186)	$\mathbf{0.64 \pm 0.2}$ (16)	$\mathbf{1.14 \pm 0.3}$ (173)
bottle_0	1.97 ± 0.4 (27)	2.26 ± 0.6 (222)	$\mathbf{1.38 \pm 0.3}$ (28)	$\mathbf{1.29 \pm 0.5}$ (221)
bottle_1	$\mathbf{0.70 \pm 0.3}$ (4)	8.79 ± 2.3 (396)	1.43 ± 0.5 (13)	$\mathbf{1.78 \pm 0.9}$ (334)

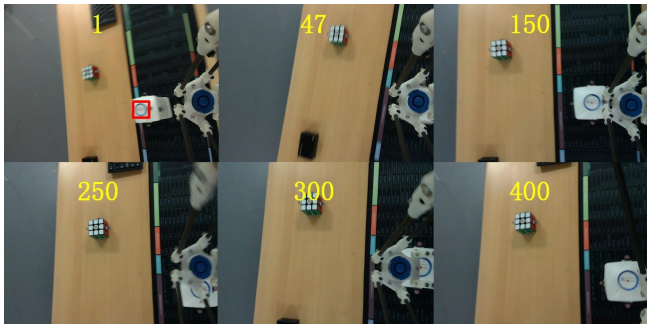

Fig. 9: Images from sequence bottle_1. The frame rate is 20 fps. The bottle cap is labelled as the target in the first frame and it is completely occluded at the 47th frame. From the 100th to the 400th frame, the MAV remains stationary, while the robotic arm is moving.

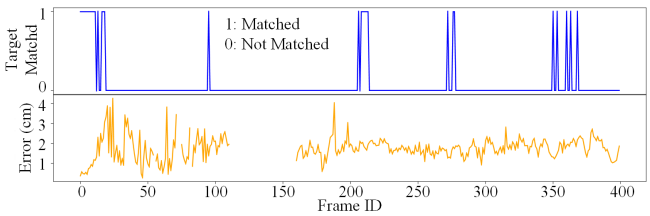

Fig. 10: Experimental results in the sequence bottle_1. The frame rate is 20 fps. The blue curve indicates whether the target point is matched in the frame, 1 denotes matched, 0 denotes not matched. The orange curve indicates the target localization error.

Since it is not the focus of our work, we use this simple method and it works well.

1) **Target localization accuracy:** The target localization accuracy of ORB-SLAM3 and our approach is shown in Table II. It also indicates the target localization error of ORB-SLAM3 is larger. Our approach, however, achieves high target localization accuracy, no matter whether the target point is matched or not.

A sample sequence (i.e., bottle_1) is shown in Fig. 9. In the first frame, the bottle cap is set as the target. The MAV approaches the target gradually, and the target is completely occluded by the robotic arm at the 47th frame. From the 100th to the 400th frame, the MAV remains stationary, while the robotic arm is moving. The localization error is shown in Fig. 10. The target localization errors of most frames are small, but the localization fails from the 110th frame (a total number of 53 frames fail). In these frames, some mismatched points bring large errors because of the movement of the robotic arm and the motion blur in images. However, our approach can continue to process new frames and match the newly detected feature points with that from the map. Finally, it recovers back at the 160th frame and is able to localize the target in subsequent frames. Furthermore, the localization error is still relatively small. It demonstrates the robustness of our approach.

2) **Robustness to outliers:** In addition to being able to cope with short-term failures, our approach is robust to cope with small amount of outliers, which may be caused by small dynamic objects or mismatched features. In the sequence shown in Fig. 11, while the MAV is approaching the target, a bottle is rolling on the table. As shown in Fig. 12, the target

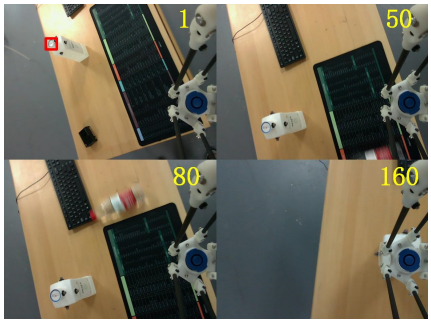

Fig. 11: Images from the sequence bottle_robustness. The bottle cap is labelled as the target in the first frame. Another bottle is rolling on the table as the MAV approaches the target.

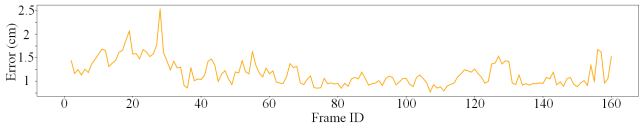

Fig. 12: Target localization error in the sequence bottle_robustness. The frame rate is 20 fps.

localization error does not become larger in this process.

The feature points belonging to the rolling bottle are also added to compute the target location, but due to their movement, the error computed according to Eq. (8) is relatively large. As described in Sec. III-D, the kernel function limits the influence of these large-error points, and they are eliminated during the optimization process. Fig. 13 shows the range-only map in this process. It indicates that the outliers from the rolling bottle have been eliminated (indicated by black points).

This experiment indicates the process of removing outliers. However, it is notable that our algorithm is designed for static scenes and may not work well with large amounts of dynamic features.

C. Run-time performance

The run-time performance of our approach is also evaluated, and the average run-time of each component is shown in Table III. Our approach takes about 42 ms (i.e., around 24 frames per second) in total to process each frame. It can be seen from Table III that the main time-consuming parts are

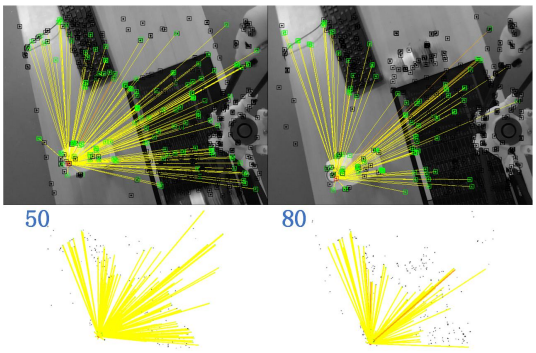

Fig. 13: Range-only map in the sequence bottle_robustness. The matched map points are drawn in green and unmatched ones are in black. The feature points belonging to the rolling bottle are eliminated, drawn in black.

TABLE III: Run-time performance

Operation	Run-time (ms)
Points Extraction	13.7
Matching with Target Point	0.22
Matching in Last Frame	0.56
Matching in Local Map	21.2
Target Localization	2.80
Total	42.1

the feature extraction and matching steps against the local map. The feature matching against the local map includes the construction of the local map, the generation of bag-of-words, and the matching of feature points, so it takes a relatively long time. The run-time can be further improved by limiting the size of the local map.

V. Conclusions

This paper proposes an approach to handle the occluded target localization problem for aerial manipulation. To achieve the task, we propose a novel localization algorithm via a target-centered range-only map. The experimental evaluations with both synthetic and real datasets demonstrate the superior performance of our proposed approach, in comparison to a state-of-the-art RGB-D SLAM-based approach.

References

[1] B. K. Hossein, J. S. Farrokh, and A. Abdelkader, "Aerial manipulation—A literature survey," *Robotics and Autonomous Systems*, vol. 107, pp. 221–235, 2018.

[2] A. Ollero and B. Siciliano, *Aerial robotic manipulation*. Springer, 2019.

[3] A. Ollero, M. Tognon, A. Suarez, D. Lee, and A. Franchi, "Past, present, and future of aerial robotic manipulators," *IEEE Transactions on Robotics*, pp. 1–20, 2021.

[4] K. Bodie, M. Brunner, M. Pantic, S. Walser, P. Pfändler, U. Angst, R. Siegwart, and J. Nieto, "An omnidirectional aerial manipulation platform for contact-based inspection," in *Robotics: Science and Systems (RSS)*, 2019.

[5] A. Suarez, G. Heredia, and A. Ollero, "Design of an anthropomorphic, compliant, and lightweight dual arm for aerial manipulation," *IEEE Access*, vol. 6, pp. 29173–29189, 2018.

[6] D. Lee, D. Jang, H. Seo, and H. Jin Kim, "Model predictive control for an aerial manipulator opening a hinged door," in *International Conference on Control, Automation and Systems (ICCAS)*, pp. 986–991, 2019.

[7] G. Garimella and M. Kobilarov, "Towards model-predictive control for aerial pick-and-place," in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4692–4697, 2015.

[8] S. Kim, H. Seo, S. Choi, and H. J. Kim, "Vision-guided aerial manipulation using a multirotor with a robotic arm," *IEEE/ASME Transactions on Mechatronics*, vol. 21, no. 4, pp. 1912–1923, 2016.

[9] H. Zhong, Z. Miao, Y. Wang, J. Mao, L. Li, H. Zhang, Y. Chen, and R. Fierro, "A practical visual servo control for aerial manipulation using a spherical projection model," *IEEE Transactions on Industrial Electronics*, vol. 67, no. 12, pp. 10564–10574, 2019.

[10] M. Laiacker, F. Huber, and K. Kondak, "High accuracy visual servoing for aerial manipulation using a 7 degrees of freedom industrial manipulator," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1631–1636, 2016.

[11] S. Zhao, Z. Hu, M. Yin, K. Z. Ang, P. Liu, F. Wang, X. Dong, F. Lin, B. M. Chen, and T. H. Lee, "A robust real-time vision system for autonomous cargo transfer by an unmanned helicopter," *IEEE Transactions on Industrial Electronics*, vol. 62, no. 2, pp. 1210–1219, 2015.

[12] Taryudi and M.-S. Wang, "3D object pose estimation using stereo vision for object manipulation system," in *International Conference on Applied System Innovation (ICASI)*, pp. 1532–1535, 2017.

[13] J. Stavnitzky and D. Capson, "Multiple camera model-based 3D visual servo," *IEEE Transactions on Robotics and Automation*, vol. 16, no. 6, pp. 732–739, 2000.

[14] Y. Yoshihata, K. Watanabe, Y. Iwatani, and K. Hashimoto, "Multi-camera visual servoing of a micro helicopter under occlusions," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2615–2620, 2007.

[15] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1309–1332, 2016.

[16] L. Fang, H. Chen, Y. Lou, Y. Li, and Y. Liu, "Visual grasping for a lightweight aerial manipulator based on NSGA-II and kinematic compensation," in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3488–3493, 2018.

[17] D. Tzoumanikas, F. Graule, Q. Yan, D. Shah, M. Popovic, and S. Leutenegger, "Aerial manipulation using hybrid force and position NMPC applied to aerial writing," in *Robotics: Science and Systems (RSS)*, 2020.

[18] Y. Luo, K. Dong, L. Zhao, Z. Sun, E. Cheng, H. Kan, C. Zhou, and B. Song, "Calibration-free monocular vision-based robot manipulations with occlusion awareness," *IEEE Access*, vol. 9, pp. 85265–85276, 2021.

[19] V. Lippiello, B. Siciliano, and L. Villani, "An occlusion prediction algorithm for visual servoing tasks in a multi-arm robotic cell," in *International Symposium on Computational Intelligence in Robotics and Automation*, pp. 733–738, 2005.

[20] D. Nicolis, M. Palumbo, A. M. Zanchettin, and P. Rocco, "Occlusion-free visual servoing for the shared autonomy teleoperation of dual-arm robots," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 796–803, 2018.

[21] K. Wada, E. Sucar, S. James, D. Lenton, and A. J. Davison, "Morefusion: Multi-object reasoning for 6D pose estimation from volumetric fusion," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14540–14549, 2020.

[22] E. Rublee, V. Rabaud, K. Konolige, and G. R. Bradski, "ORB: an efficient alternative to SIFT or SURF," in *International Conference on Computer Vision (ICCV)*, pp. 2564–2571, 2011.

[23] D. Gálvez-López and J. D. Tardós, "Bags of binary words for fast place recognition in image sequences," *IEEE Transactions on Robotics*, vol. 28, no. 5, pp. 1188–1197, 2012.

[24] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "g2o: A general framework for graph optimization," in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3607–3613, 2011.

[25] P. J. Huber, "Robust estimation of a location parameter," in *Breakthroughs in Statistics*, pp. 492–518, Springer, 1992.

[26] A. Handa, T. Whelan, J. McDonald, and A. J. Davison, "A benchmark for RGB-D visual odometry, 3D reconstruction and SLAM," in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1524–1531, 2014.

[27] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. M. Montiel, and J. D. Tardós, "Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam," *IEEE Transactions on Robotics*, pp. 1–17, 2021.

[28] F. Furrer, M. Fehr, T. Novkovic, H. Sommer, I. Gilitschenski, and R. Siegwart, "Evaluation of combined time-offset estimation and hand-eye calibration on robotic datasets," in *Field and Service Robotics*, pp. 145–159, 2018.