Selective Placement of Visual Fiducials for Localization Tasks

Jaimie Carlson[1], Andreas Kolling[1]

***Abstract*— This paper describes how to define and execute tasks that depend on localization for their success. Real-world robotic systems that perform precise work at endpoints generally have tasks that fail if a robot's localization error is above the task requirement. However, most systems for considering localization error are task-agnostic. We distinguish between business-case-required *work tasks* and *localization tasks*, which bound localization errors, e.g., by requiring a robot to perceive visual fiducial tags. We describe how to sequence these tasks and define localization tasks to decrease their probability of failure. We identify localization requirements for a case study of using visual fiducials to make those requirements less stringent, and optimize placement of visual fiducials to make the system robust to failures due to localization.**

I. Introduction

Localization is a fundamental capability of autonomous robots, and it supports much of the physical work a robot executes. Unsurprisingly, errors in pose estimates can cause failures in task execution. Commercial systems hence benefit from clear requirements for the performance of a localization system, but this poses a number of challenges. First, most SLAM approaches use probabilistic methods and are designed to operate robustly in conditions that are not formalized, making formal analysis difficult. Second, most SLAM approaches also work incredibly well most of the time, hiding the long tail of rare errors that are relevant for cost-effective commercial deployment at scale, with hundreds of thousands of robots. Third, it is difficult to predict the impact on localization and task success of a given change to the environment, such as installing a visual fiducial in a specific place, and balance the maintenance burden of such changes.

The performance of localization systems is often benchmarked by measuring the localization error. One localization error definition is the Absolute Trajectory Error (ATE), which measures the distance between estimated poses $\hat{x}$ and ground truth poses x, i.e., $ATE := ||\hat{x} - x||$. Many authors, e.g., [1], [2], and [3], evaluate a localization system's error by taking the RMSE of ATE across discrete ground-truth poses $(x_i)_{i=0}^N$ and their corresponding estimated poses $(\hat{x}_i)_{i=0}^N$.

$$RMSE = \sqrt{\sum_{i=0}^{N} ||\hat{x}_i - x_i||^2} \tag{1}$$

Relative Pose Error (RPE) focuses on local consistency when estimated and ground truth poses are estimated in different frames; it individually aligns small segments of (x_i) and $(\hat{x}_i)$, then compares their offsets [4]. There are other methods of benchmarking SLAM algorithms found in [5] and [6] (a thorough review is in [7]).

Some metrics are more task-specific: Overlap Displacement Error (ODE) uses (x_i), $(\hat{x}_i)$, and a robot's sensor frustum to calculate map consistency as required by navigation tasks [8]. Correct Rate (CR), the amount of the states in a trajectory which have below a certain required translation and rotation error, can be made task-specific [9].

However, all these methods are either task-agnostic, or require knowledge of x, which presents the following issues. First, in a deployed production system, it is rarely feasible to acquire x, since GPS (used in [10]) is not usable indoors, and motion-capture systems (used in [2], [11]) are expensive and impractical to install in active work environments. Second, localization performance error metrics should include task-dependent context for systems that accomplish work. For instance, a very small error is needed to properly dock with a charger outlet. However, a larger error is acceptable if navigating through a wide-open area. Thus, localization error metrics should not have one requirement for all tasks.

II. Problem Definition

Our problem consists of defining localization requirements: under what conditions will a robot's localization error directly cause a task to fail? We limit our system to robots moving on a flat plane with poses $x \in SE(2)$ with a localization system that produces pose estimates $\hat{x}$. We first introduce pose sets to avoid reasoning about distributions and then define tasks in relation to these pose sets.

Definition 1: Define $\hat{X}^\epsilon \subset SE(2)$ and call it an ϵ-pose-set, where, with probability ϵ, the robot's actual pose $x \in \hat{X}^\epsilon$:

$$\epsilon = P(x \in \hat{X}^\epsilon) = \int_{x' \in \hat{X}^\epsilon} P(x = x')dx' \tag{2}$$

Note that there could be multiple $\hat{X}^\epsilon$ for any given random variable P for which $\hat{x}$ is an estimate. For simplicity, we shall construct one $\hat{X}^\epsilon$ for a given $\hat{x}$. In this case $\hat{X}^\epsilon$ could correspond to a 100ϵ-% confidence interval for the Gaussian distribution $\mathcal{N}(\hat{x}, \mathbf{P})$, where $\hat{x}$ is the mean and $\mathbf{P}$ is some covariance $\text{diag}(\sigma_x, \sigma_y, \sigma_\theta)$.

Definition 2: A task τ is defined as a tuple:

$$\tau = (x^*, S, F) \tag{3}$$

where $x^* \in SE(2)$ is the goal pose which the robot is expected to reach for the task. We ignore errors in the robot's actuation and motion planning and assume that the robot has moved its estimated pose perfectly to the goal pose at the start of a task: $\hat{x} = x^*$.

[1] Amazon Global Robotics carjaimi, kollinga@amazon.com

$S \subset SE(2)$ is the success set: if the robot's pose $x \in S$, then the task's localization requirements are met and the task can execute successfully. Otherwise, the localization requirements are considered not met and the task may fail. S is motivated by the task-space region (TSR) from [12], but less constrained; in [12] TSRs are defined by lower and upper bounds on (x, y, z, roll, pitch, yaw) but success sets can be any set of poses, seen in Fig. 1 and 3. Note that the robot only knows $\hat{x}$ and cannot know x without a perfect localization system. In addition, tasks can fail for other reasons than localization. Therefore, the robot determines actual task success with an external process, such as the robot's successful docking with a charger, or perception of a specific endpoint target.

Finally, F models how a robot goes through a finite sequence of poses $(x_i)_{i=0}^N$, with associated estimated poses $(\hat{x}_i)_{i=0}^N$ while executing a task. A task starts at pose x_0 and ends at x_N. For each $\hat{x}_i$ we have a corresponding $\hat{X}_i^\epsilon$. We call F the pose-set-modifier and it describes how the ϵ-pose-sets change from the beginning to the end of a task:

$$F : \hat{X}_0^\epsilon \mapsto \hat{X}_N^\epsilon \tag{4}$$

We now introduce three special types of tasks: work tasks, localization tasks, and navigation tasks with respective subscripts τ_w, τ_ℓ and τ_n.

First, work tasks are defined by some external pose constraint: i.e., the need for a robot to dock with a payload, a charger, or some other fixed location.

Definition 3: A work task is defined as a tuple:

$$\tau_w = (x_w^*, S_w, F_w = ID) \tag{5}$$

where ID is the identity, i.e., pose estimates do not change throughout the course of a work task.

Fig. 1 illustrates τ_w with the example of parking a car: the goal pose x_w^* is at the center of the spot, and S_w is defined as any pose which places the car in the parking lines.

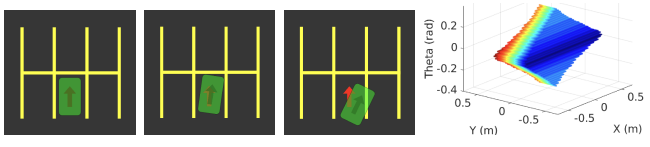

Fig. 1. A robot reaching a goal pose $x^* = \hat{x}$ (red arrow) with three localization errors from left to right. The third case fails because the error is too large, exceeding S. The rightmost plot shows the corresponding S for which all corners of a 1.5m x 4.5m car are inside a 3x5m parking space. Color illustrates Y-dimension.

Second, localization tasks are used to improve localization in order to perform work tasks. They are intermediate tasks whose goal poses can be adjusted, since they are are not defined by a business use-case. For instance, a localization task can represent a robot perceiving a visual fiducial.

Definition 4: A localization task is defined as a tuple:

$$\tau_\ell = (x_\ell^*, S_\ell, F_\ell, f_\ell, r_\ell) \tag{6}$$

The robot does not move, i.e., $x_0 = x_N$, but $N = 1$, and F_ℓ can modify $\hat{X}_0^\epsilon$ to $\hat{X}_N^\epsilon$. In addition to goal pose x_ℓ^* and success set S_ℓ, localization tasks have a pose-observation function f_ℓ, which outputs a pose estimate $f(x_0) = \hat{x}_{obs}$ observed from the robot's actual pose x_0.

$$f_\ell : S_\ell \to SE(2) \tag{7}$$

f_ℓ is defined only on S_ℓ but it may return pose observations outside of S_ℓ. Generally, these are expected to be close by. If $x_0 \in S_\ell$ and the process producing pose observations returns successfully, then τ_ℓ is successful and it guarantees that:

$$||x_0 - \hat{x}_{obs}|| < r_\ell \tag{8}$$

In colloquial terms, reaching S_ℓ for a localization task produces a pose estimate $\hat{x}_{obs}$ with an error smaller than r_ℓ. This estimate is integrated into a localization system. For simplicity, we assume that we set $\hat{x}_N := \hat{x}_{obs}$ if τ_ℓ succeeds thereby guaranteeing that $||x_N - \hat{x}_N|| < r_\ell$ and therefore bounding the size of $\hat{X}_N^\epsilon$. In practice, for small r_ℓ we expect that $\hat{X}_N^\epsilon$ is smaller than $\hat{X}_0^\epsilon$, i.e., the observation not only improved our estimate, it also reduced our variance which we reason about with the set of possible poses.

Note that in practice one may have $x_0 \notin S_\ell$ but the observation process may still succeed and produce an estimate. We ignore this complication.

Third, we introduce a navigation task to model how a robot's pose estimates change when moving between tasks with different goal poses.

Definition 5: A navigation task is defined as a tuple:

$$\tau_n = (x_n^*, \hat{x}_N, S_n = SE(2), F_n) \tag{9}$$

Navigation tasks are considered to always meet the localization requirements for success, hence $S_n = SE(2)$.

Note that x_n^* is the goal pose for beginning the navigation task and therefore the start pose $\hat{x}_0$ for navigation. At the end of a navigation task, a robot will have estimated pose at the target pose $\hat{x}_N$, real pose x_N, and probable pose set $\hat{X}_N^\epsilon$. We assume that navigation is achieved by some controller which generates $(x_i)_{i=0}^N$ and therefore F_n in combination with the localization system. The main purpose of navigation tasks is to model how much $\hat{X}^\epsilon$ will grow from $\hat{X}_0^\epsilon$ to $\hat{X}_N^\epsilon$ while moving. An implementation of this might involve using filtering techniques to predict how much the covariance of a robot's pose estimate will grow, and modeling this in F_n.

III. Making Tasks More Achievable

Now that we have defined tasks and can determine whether a robot's state and ϵ-pose-set meet task localization requirements, we can specify and order tasks to decrease the probability of failure. Consider a sequence of tasks $(\tau_i)_{i=0}^M$. If the robot executed task τ_i, is at its estimated position $\hat{x}_N$, and needs to complete a next task τ_{i+1}, the task may fail if ϵ-pose set $\hat{X}_N^\epsilon \not\subset S_{i+1}$, but will meet the localization requirements with probability $\geq \epsilon$ if $\hat{X}_N^\epsilon \subset S_{i+1}$. For convenience, we shall focus on a simple task sequence of only $\tau_n, \tau_\ell, \tau_w$. Here it is best to think of τ_n as introducing error and widening the pose set but allowing the robot to move, and τ_ℓ as having a wide S_ℓ and a small r_ℓ which matches the corresponding

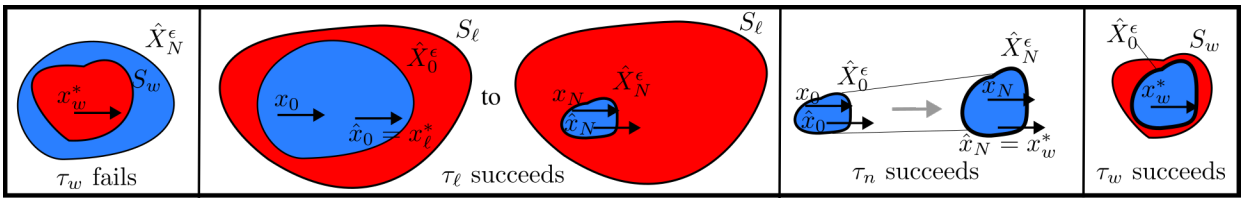

Fig. 2. (a) $\hat{X}_N^\epsilon \not\subset S_w$, so work task may fail. (b) Localization task succeeds because $\hat{X}_0^\epsilon \subset S_\ell$, shrinking the ϵ-pose-set to $\hat{X}_N^\epsilon$ for the next task. (c) A navigation task moves $\hat{x}$ to x_w^*, but grows $\hat{X}^\epsilon$. (d) $\hat{X}_0^\epsilon \subset S_w$, so work task is now guaranteed to succeed with probability ϵ.

size of S_w. The basic pattern of $\tau_n, \tau_\ell, \tau_w$ is similar to the idea of image pre-chaining, as defined in [13] and [14].

In practice, work tasks are pre-defined and fixed. Navigation tasks derive from the need to place the robot at the goal poses of work tasks, and appropriate localization tasks are added to make sure $\hat{X}_N \subset S_{i+1}$ holds true for the sequence.

For example, consider a robot with a current pose at $\hat{x} = x_w^*$, which has to complete a work task τ_w. However, its localization uncertainty is too high, so its possible pose set, $\hat{X}^\epsilon$, is not a subset of S_w (Fig. 2a). The task could be pre-chained with a localization task. If $\exists \tau_\ell$ s.t. $\hat{X}^\epsilon \subset S_\ell$, and $F_\ell(\hat{X}^\epsilon) \subset S_w$, task chain $[\tau_\ell, \tau_w]$ will succeed. Alternately, the task could also be pre-chained with a navigation task. If $\exists \tau_n$ s.t. $F_n(\hat{X}^\epsilon) \subset S_w$, task chain $[\tau_n, \tau_w]$ will succeed. This is illustrated in Fig. 2: a work task is originally not guaranteed to succeed because it has too much of a pose error. A localization task at the same pose shrinks the ϵ-pose-set. Then, a navigation task is performed to bring the new pose estimate to the goal x_w^*. This grows the ϵ-pose-set, but it is still a subset of S_w, so the work task can succeed.

There are several ways to exploit this formalization. One is to start with a parametrization of localization tasks. For instance, a landmark or fiducial in an environment may provide a pose observation whose quality depends on the distance to it. In this case, by modifying r_ℓ and observing how S_ℓ shrinks or grows, multiple tasks can be created from the same landmark. This can allow a planner to generate the localization task with the appropriate r_ℓ dynamically from available landmarks or fiducials. Similarly, the goal poses can be adjusted, leading to a different F_ℓ for a given fiducial.

Alternatively, one can also hand-craft task sequences and use the formalization to evaluate whether these succeed. This still requires an experimental characterization of F_ℓ and F_n.

In addition to pre-chaining tasks, we can make tasks inherently easier to achieve: informally, increasing the size of S. For the work task in Fig. 1, this could mean making a parking space larger, or a car smaller, so it is easier to fit the car in the space. For a localization task, as the one shown in Fig. 3, a visual fiducial could be made larger, the lighting around it improved, or its shape optimized to decrease pose error (as in [15]), so that an accurate pose estimate can be acquired from further away.

Third, we can better place S with respect to x^* to ensure $\hat{X}^\epsilon \subset S$. In the next section, we do this for the use case where τ_ℓ requires viewing a planar visual fiducial.

IV. Real-World Use Case

This section applies our task framework to an example use case run on a robot with commercially available hardware. It programmatically defines $\hat{X}^\epsilon$ for its localization system and S for its localization task of viewing a visual fiducial. It describes how to optimize the visual fiducial's pose to maximize the chance $\hat{X}^\epsilon \subset S$ for the localization task.

A. Setup

This system uses a robot with a forward-facing Realsense camera used to read large visual fiducials on the walls (with forward-camera to robot transform $T_c^r \in SE(3)$).

In our example, our work task τ_w requires that a robot must navigate to an endpoint and do some precise work, for which S_w is quite small (as shown by the bounding box $TSR_w \subset S_w$ in Fig. 3a, which extends a few centimeters in x and y dimensions, and a few degrees in θ dimension). Assume that during normal operation of this robot's SLAM system, $\hat{X}^\epsilon$ is centered around the SLAM pose estimate $\hat{x}$ with < 1 m translation error and $< 2°$ rotation error. So, $||x - \hat{x}||$ is often higher than several centimeters, meaning $\hat{X}^\epsilon \not\subset S_w$ when approaching x_w^* and the task can fail, as shown in Fig. 3b. This causes frequent failures of task τ_w, which prevents consistent system operation.

To alleviate such problems, we pre-chained τ_w with a τ_ℓ, in which a visual fiducial (April Tag [16]) was placed with tag pose in world frame $T_t^w \in SE(3)$. If the robot got a "good detection" of the tag, the pose estimates from the tag detections were added into the robot's localization estimate. For a "good detection" $x \in S_\ell$, the robot's pose in the world frame $T_r^w \in SE(3)$ must follow certain conditions.

First, the visual fiducial is detected. For the tag to appear in the field of view of the robot's front camera, $T_r^w \in \mathcal{V}$, the visibility region of the tag: the set of robot poses from which all 4 tag corners (u, v) appear in the camera's field of view, as calculated by projecting tag points into the camera's 2D image plane via the projection equation.

Second, the camera-to-tag distance and angle are low enough to yield a good pose estimate. As discussed in [17], pose estimate error bounds can be fit based on camera-to-tag transform T_c^t: they increase with camera-to-tag distance d, angle from camera normal to camera-to-tag line of sight ϕ, and angle from tag normal to camera-to-tag line of sight ψ.

We define S_ℓ for our task (shown in Fig. 3c) as the T_r^w where $T_r^w \in \mathcal{V}$ and where $T_c^t = T_w^t \times T_r^w \times T_c^r$ has $d < d_{max}$, $\phi < \phi_{max}$, and $\psi < \psi_{max}$, such that tag estimate error $||\hat{x}_{obs} - x||$ < a certain threshold r_ℓ.

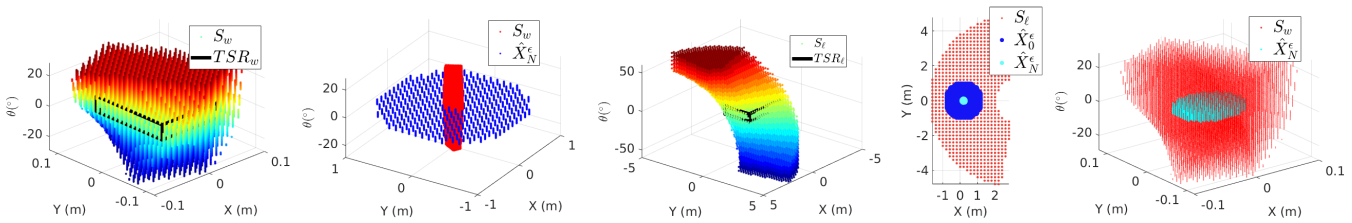

Fig. 3. Red indicates S, blue indicates $\hat{X}$. For (a) and (c), color illustrates theta dimension and bounding boxes $TSR \subset S$ for illustrative purposes. $x^*_w = x^*_\ell$ = (0m, 0m, 0°). **(a)** The work task can be done from the small set of poses S_w. **(b)** The large starting ϵ-pose-set $\hat{X}^\epsilon_N \not\subset S_w$, so τ_w may fail. **(c)** S_ℓ is the set of poses from which the robot can detect an April Tag with pose error $< r_\ell$. **(d)** $\hat{X}^\epsilon_0 \subset S_\ell$, so τ_ℓ must succeed - this decreases $\hat{X}^\epsilon_0$ to the smaller $\hat{X}^\epsilon_N$, where $||x' - x|| < r_\ell \; \forall x' \in \hat{X}^\epsilon_N$. **(e)** $\hat{X}^\epsilon_N$ for τ_ℓ = $\hat{X}^\epsilon_N$ for next task τ_w. $\hat{X}^\epsilon_N \subset S_w$, so τ_w can be achieved.

So, $\hat{X}^\epsilon \subset TSR_\ell \subset S_\ell$, so $\hat{X}^\epsilon \subset S_\ell$, and the pre-chained localization task τ_ℓ will succeed (Fig. 3c). Then, after April Tag pose estimates decrease $||x - \hat{x}|| \leq r_\ell$, $\hat{X}^\epsilon$ is centered around a new estimated pose $\hat{x}$ with < 5 cm translation error and $< 2°$ rotation error (Fig. 3d).

Then, if $\hat{X}^\epsilon \not\subset X_w$, a navigation task $\tau_n = (\hat{x}, x^*_w, F_n)$ can be specified to move the task from $\hat{x}$ to x^*_w. Because the robot remains in the field of view of the tag, it is expected that the uncertainty will not grow very much, $\hat{X}^\epsilon = F_n(\hat{X}^\epsilon) \subset S_w$, so the task can be completed (Fig. 3e).

B. Visual Fiducial Placement Cost Function

For reasons of cost, visual fiducials cannot be placed all around a building; they should be placed selectively near work endpoints with stringent localization requirements.

The problem is to place an tag with some transform $T^t_e \in SE(3)$ between the tag and a work endpoint which has pose $T^w_e \in SE(3)$. We define a cost function to maximize the chance that the tag is in the drive's camera's field of view and minimize pose error. The cost c of viewing a tag from a given robot pose x or T^w_r, with $T^t_r = T^t_e \times T^e_w \times T^w_r$ and associated d, ϕ, ψ calculated from $T^t_c = T^t_r \times T^r_c$ is:

$$c = \begin{cases} c_{high} & \text{if } T^w_r \notin \mathcal{V},\ d > d_{max}, \\ & \phi > \phi_{max}, \text{ or } \psi > \psi_{max} \\ err(d, \phi, \psi) & \text{if } T^w_r \in \mathcal{V} \end{cases} \quad (10)$$

$$err(d, \phi, \psi) = c_1 d^2 + c_2 \phi^2 + c_3 \psi^2 \quad (11)$$

where c_{high}, c_1, c_2, and c_3 are experimentally-set constants.

For fixed T^w_e and T^r_c, find a T^t_e that minimizes expected cost $E[c]$ over all robot poses x in the pose set $\hat{X}^\epsilon$ with $P(x = x')$ sampled from the normal distribution $\mathcal{N}(x^*, \mathbf{P})$:

$$E[c] = \int_{x' \in \hat{X}^\epsilon} P(x = x') c(x', T^t_e) dx' \quad (12)$$

C. Visual Fiducial Placement Results

Matlab's genetic algorithm implementation *ga* was used to find the T^t_e that minimizes c. For practicality's sake, the search space constrained the roll and pitch of T^t_e to 0 (i.e., the tag is hanging vertically on a wall).

Results are shown in Fig. 4. When we assume $x = \hat{x}$, the tag is placed as close as possible to the endpoint pose; when a larger $\hat{X}^\epsilon$ is assumed, the tag is placed further from the endpoint pose. This could lead to increased observation pose error in the case where $x = \hat{x}$, but in the average case

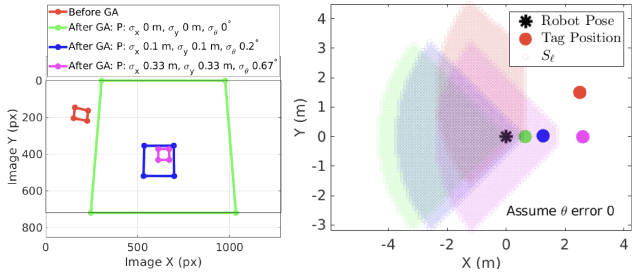

Fig. 4. $c_{high} = 1e6$, $c_1 = 100$, $c_2 = 50$ and $c_3 = 50$. Before *ga* (red), the tag is placed far away and at an angle from the robot, with the robot at the endpoint pose on the border of its visibility region, potentially causing failure to get a detection. With *ga* (green, blue, pink), the tag is centered in the image. (a) Reprojection of the tag points onto the image plane. As $\hat{X}$ grows (green → blue → pink), the tag appears smaller. (b) Top-down view in the world frame of S_ℓ for each tag pose. As $\hat{X}$ grows, distance from robot pose to tag position, and distance from robot pose to S_ℓ boundary, increase.

increases the chance that the robot will see the tag, by placing the robot's expected pose at the endpoint farther from the boundary of S_ℓ, meaning that a larger $\hat{X}^\epsilon \subset S_\ell$.

We can use this problem to inform the placement of tags at endpoints in a deployed system. In real scenarios, T^t_e may be constrained, due to the necessity of integrating into a working site with pre-existing equipment. In those cases, this cost function provides a mechanism to compare $E[c]$ between different candidate locations, and choose the tag pose that best helps the robot improve its localization.

V. Conclusion

In this paper, we have presented a formulation to describe tasks that depend on localization to succeed. By comparing the robot's current ϵ-pose-set $\hat{X}^\epsilon$ and the success set S for a task, we can determine whether the robot meets the task's localization requirements. We show an example where a work task's completion could not be guaranteed due to high pose error; it was pre-chained with an achievable localization task that decreased its pose error so it could be completed. Finally, that localization task was made more robust by optimizing the placement of a visual fiducial to maximize the chance it was in the robot's field of view while minimizing observation pose error. Future work will present an algorithm for defining and pre-chaining localization and navigation tasks to provide guarantees on success for a given work task.

VI. Acknowledgement

Thanks to Amanda Adkins and Chris Keane for comments and guidance.

References

[1] Y. Liu and J. Miura, "Rds-slam: Real-time dynamic slam using semantic segmentation methods," *IEEE Access*, vol. 9, pp. 23 772–23 785, 2021, ISSN: 21693536. DOI: 10.1109/ACCESS.2021.3050617.

[2] R. Mur-Artal, J. M. Montiel, and J. D. Tardos, "Orb-slam: A versatile and accurate monocular slam system," *IEEE Transactions on Robotics*, vol. 31, 5 2015, ISSN: 15523098. DOI: 10.1109/TRO.2015.2463671.

[3] A. Rosinol, A. Violette, M. Abate, *et al.*, "Kimera: From slam to spatial perception with 3d dynamic scene graphs," *International Journal of Robotics Research*, vol. 40, 12-14 2021, ISSN: 17413176. DOI: 10.1177/02783649211056674.

[4] R. Kümmerle, B. Steder, C. Dornhege, *et al.*, "On measuring the accuracy of slam algorithms," *Autonomous Robots*, vol. 27, 4 2009, ISSN: 09295593. DOI: 10.1007/s10514-009-9155-6.

[5] Y. Bar-Shalom, X.-R. Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation*. 2001. DOI: 10.1002/0471221279.

[6] X. S. Le, L. Fabresse, N. Bouraqadi, and G. Lozenguez, "Evaluation of out-of-the-box ros 2d slams for autonomous exploration of unknown indoor environments," vol. 10985 LNAI, 2018. DOI: 10.1007/978-3-319-97589-4_24.

[7] Z. Zhang and D. Scaramuzza, "A tutorial on quantitative trajectory evaluation for visual(-inertial) odometry," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 7244–7251. DOI: 10.1109/IROS.2018.8593941.

[8] C. Mostegel, J. Ye, Y. Luo, and Y. Liu, "Overlap displacement error: Are your slam poses map-consistent?" In *IROS 2021*, 2021. [Online]. Available: https://www.amazon.science/publications/overlap-displacement-error-are-your-slam-poses-map-consistent.

[9] X. Shi, D. Li, P. Zhao, *et al.*, "Are we ready for service robots? the openloris-scene datasets for lifelong SLAM," *CoRR*, vol. abs/1911.05603, 2019. arXiv: 1911.05603. [Online]. Available: http://arxiv.org/abs/1911.05603.

[10] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *International Journal of Robotics Research*, vol. 32, 11 2013, ISSN: 02783649. DOI: 10.1177/0278364913491297.

[11] H. Sier, L. Qingqing, Y. Xianjia, J. P. Queralta, Z. Zou, and T. Westerlund, *A benchmark for multi-modal lidar slam with ground truth in gnss-denied environments*, 2022. DOI: 10.48550/ARXIV.2210.00812. [Online]. Available: https://arxiv.org/abs/2210.00812.

[12] D. Berenson, S. S. Srinivasa, and J. J. Kuffner, "Addressing pose uncertainty in manipulation planning using task space regions," 2009. DOI: 10.1109/IROS.2009.5354833.

[13] L. P. Kaelbling and T. Lozano-Pérez, "Pre-image backchaining in belief space for mobile manipulation," in *Robotics research*, Springer, 2017, pp. 383–400.

[14] T. Lozano-Pérez, M. T. Mason, and R. H. Taylor, "Automatic synthesis of fine-motion strategies for robots," *The International Journal of Robotics Research*, vol. 3, 1 1984, ISSN: 17413176. DOI: 10.1177/027836498400300101.

[15] J.-K. Huang, W. Clark, and J. W. Grizzle, "Optimal target shape for lidar pose estimation," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 1238–1245, 2022. DOI: 10.1109/LRA.2021.3138779.

[16] E. Olson, "Apriltag: A robust and flexible visual fiducial system," 2011. DOI: 10.1109/ICRA.2011.5979561.

[17] S. M. Abbas, S. Aslam, K. Berns, and A. Muhammad, "Analysis and improvements in apriltag based state estimation," *Sensors (Switzerland)*, vol. 19, 24 2019, ISSN: 14248220. DOI: 10.3390/s19245480.